Πανεπιστήμιο Θεσσαλίας
Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων (ΤΜΗΥΤΔ)

Διπλωματική εργασία

ΤΙΤΛΟΣ : Ιδιωτικότητα στην Εξόρυξη Γνώσης Τροχιών Κινουμένων Αντικειμένων

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ : Βερύκιος Βασίλειος

Πετράκη Ελένη

Βόλος, 2009

Περιεχόμενα

ΚΕΦΑΛΑΙΟ 1

1. Εισαγωγή

Οι εξελίξεις στις τηλεπικοινωνίες και στις τεχνολογίες πληροφόρησης, καθιστούν αναγκαία την ανάπτυξη αποδοτικών, υψηλής ποιότητας υπηρεσιών που βασίζονται στη θέση των χρηστών που τις χρησιμοποιούν (LBS: Location Based Services). Οι θέσεις από τις οποίες διέρχεται κάθε χρήστης συλλέγονται από ειδικά συστήματα (GPS, radio-locationing, RFID) που είναι ενσωματωμένα σε διάφορες συσκευές (κινητά τηλέφωνα, PDAs, GPS devices). Όταν ένας χρήστης ζητά την παροχή κάποιας LBS υπηρεσίας, στέλνει μέσω της συσκευής του την τρέχουσα θέση του σε έναν απομακρυσμένο παροχέα της υπηρεσίας, που βρίσκεται εγκατεστημένος στον τηλεπικοινωνιακό φορέα στον οποίο είναι εγγεγραμμένος ο χρήστης. Επειδή όμως υπάρχει πιθανότητα ο παροχέας να είναι μη ασφαλής, ο χρήστης κατά την αποστολή της αίτησής του αποφεύγει να στείλει ευαίσθητα προσωπικά δεδομένα, που θα έθεταν σε κίνδυνο την ιδιωτικότητά του. Μπορεί, παραδείγματος χάριν, ένας χρήστης μιας συσκευής GPS να στείλει μια αίτηση στον παροχέα, με την οποία θα ζητά να μάθει ποιο φαρμακείο εφημερεύει στην περιοχή από την οποία έκανε την αίτηση χωρίς να θέλει να αποκαλυφθεί το όνομά του. Δημιουργούνται έτσι δύο αλληλοσυγκρουόμενα ζητήματα. Από τη μια πλευρά ο παροχέας πρέπει να γνωρίζει τα ακριβή γεωγραφικά και χρονικά στοιχεία της αίτησης, ενώ από την άλλη η ταυτότητα του χρήστη πρέπει να προστατευθεί. Οι πιθανότητες να αποκαλυφθούν στοιχεία για την ταυτότητα ενός χρήστη, αυξάνονται όταν ο τελευταίος ακολουθεί συχνά μια διαδρομή (trajectory) ή ένα τμήμα μιας διαδρομής (subtrajectory).

Στην παρούσα εργασία, προκειμένου να προστατευθεί η ιδιωτικότητα των χρηστών και να ανακαλυφθούν πιο ρεαλιστικά πρότυπα κίνησης, προτείνεται μια βελτιωμένη αναπαράσταση και μια καινούρια τεχνική εξόρυξης των συχνών διαδρομών που εκτελεί κάθε χρήστης (frequent subtrajectories). Θεωρούμε ότι οι χρήστες κινούνται σε μια «εικονική» πόλη, που δημιουργήσαμε με ειδικό πρόγραμμα (Generator του Brinkhoff) και οι διαδρομές αποθηκεύονται σε μια χώρο – χρονική Βάση Δεδομένων στο Σ.Δ.Β.Δ. της Oracle.

Η δομή της παρούσας εργασίας είναι η εξής: στο Κεφάλαιο 2 θα παρουσιάσουμε τη σχετική έρευνα που έχει γίνει στον τομέα της εξόρυξης γνώσης από χώρο – χρονικές Β.Δ. και πιο συγκεκριμένα για την εύρεση συχνών διαδρομών και θα αναφερθούν τα πλεονεκτήματα και τα μειονεκτήματα των τεχνικών που έχουν προταθεί. Στο Κεφάλαιο 3 θα περιγράψουμε τον τρόπο με τον οποίο παράγουμε χώρο - χρονικά δεδομένα και πώς γίνεται η εισαγωγή τους στο δίκτυο. Στο Κεφάλαιο 4 θα προτείνουμε έναν τρόπο αναπαράστασης των διαδρομών στη Β.Δ. Στο Κεφάλαιο 5 θα περιγράψουμε και θα υλοποιήσουμε την τεχνική με την οποία βρίσκουμε συχνές διαδρομές. Στο Κεφάλαιο 6 περιγράφεται σε θεωρητικό υπόβαθρο το πρόβλημα προστασίας της ιδιωτικότητας χώρο – χρονικών δεδομένων. Στο Κεφάλαιο 7 θα κάνουμε πειραματικές μετρήσεις ώστε να αξιολογήσουμε την τεχνική που θα υλοποιήσουμε και να τη συγκρίνουμε με τεχνικές από προηγούμενες σχετικές εργασίες.

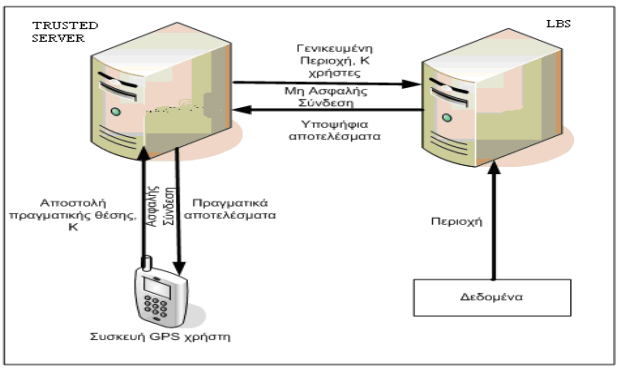
ΚΕΦΑΛΑΙΟ 2

2. Σχετική έρευνα

Στον τομέα προστασίας της ιδιωτικότητας των χρηστών, που χρησιμοποιούν υπηρεσίες βασισμένες στην τοποθεσία (LBS), έχουν προταθεί διάφορες τεχνικές, που στηρίζονται στην εύρεση συχνών διαδρομών που ακολουθεί ο χρήστης. Μία από αυτές είναι και η K-ανωνυμία. Στόχος της τεχνικής αυτής είναι η προστασία δεδομένων που αφορούν την ιδιωτική ζωή κάθε χρήστη, του οποίου τα στοιχεία είναι αποθηκευμένα σε μια Βάση Δεδομένων. Η τεχνική αυτή προτάθηκε αρχικά για σχεσιακές ΒΔ από τις Samarati και Sweeney στο [4]. Στα [2] και [3] η τεχνική της K-ανωνυμίας εφαρμόστηκε από τους Σ.Γιαννακόπουλο και Δ.Δούρα αντίστοιχα για χώρο-χρονικές ΒΔ, που αποθηκεύονται στο ΣΔΒΔ της Oracle. Με βάση τον ορισμό που δίνεται στο [3], στις LBS υπηρεσίες ένας χρήστης είναι K-Ανώνυμος σε μια περιοχή, αν και μόνο αν υπάρχουν τουλάχιστο k-1 άλλοι χρήστες στην περιοχή αυτή, την ίδια χρονική στιγμή.

2.1 Γενικό μοντέλο LBS

Οι περισσότερες τεχνικές που υλοποιούν την K-ανωνυμία σε LBS χρησιμοποιούν το γενικό μοντέλο που φαίνεται στο Σχήμα 2.1.



Σχήμα 2.1 : Γενικό μοντέλο LBS

Σύμφωνα με το μοντέλο αυτό, υπάρχει ένας κεντρικός εξυπηρετητής, που παρέχει την LBS υπηρεσία, με τον οποίο επικοινωνεί ένας ενδιάμεσος έμπιστος εξυπηρετητής (Trusted Server – TS) και όχι ο ίδιος ο χρήστης. Ο χρήστης ζητά την παροχή μιας υπηρεσίας στέλνοντας την αίτηση και την πραγματική του θέση στον TS. Ο TS αφαιρεί το αναγνωριστικό του χρήστη και μετατρέπει τη συγκεκριμένη θέση του χρήστη σε μια γενικευμένη περιοχή στην οποία βρίσκονται άλλοι κ-1 χρήστες. Έπειτα στέλνει την αίτηση με τη γενικευμένη περιοχή στον κεντρικό εξυπηρετητή και ο τελευταίος, αφού την επεξεργαστεί, επιστρέφει στον TS ένα σύνολο υποψήφιων αποτελεσμάτων στα οποία εμπεριέχονται τα πραγματικά. Τέλος, ο TS, αφού γνωρίζει την πραγματική θέση του αιτούντα, στέλνει σε αυτόν τα πραγματικά αποτελέσματα.

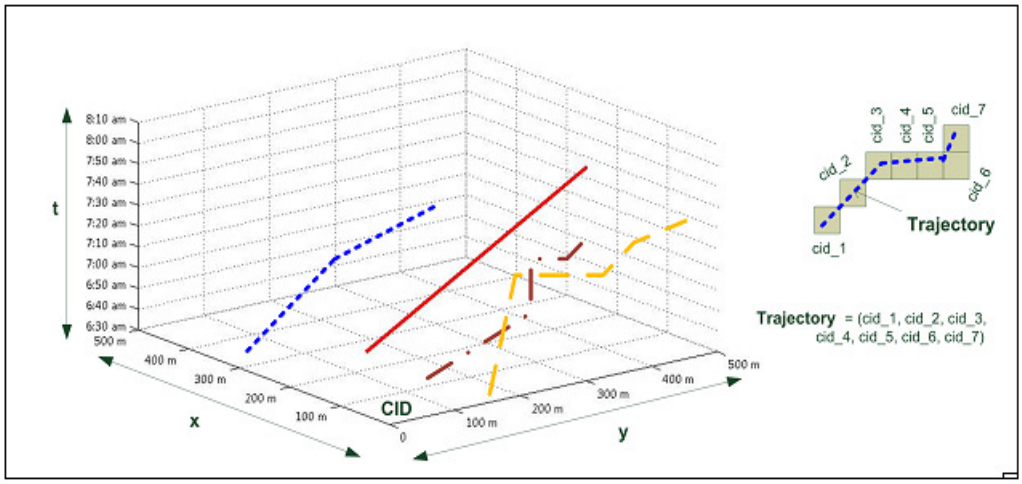
2.2 Προσδιοριστές προστασίας

Η τεχνική που πρότειναν οι Samarati και Sweeney, βασίζεται στους προσδιοριστές προστασίας, οι οποίοι ορίζονται και περιγράφονται στο [5]. Με βάση αυτόν τον ορισμό, προσδιοριστής προστασίας είναι το σύνολο των γνωρισμάτων της βάσης, τα οποία σε συνδυασμό με πληροφορίες από εξωγενείς παράγοντες μπορούν να

προσδιορίσουν την ταυτότητα των χρηστών. Ο ορισμός αυτός επεκτάθηκε και για χώρο-χρονικές ΒΔ. Ενώ στις περισσότερες τεχνικές K-ανωνυμίας σε χώρο-χρονικές ΒΔ, ως προσδιοριστής προστασίας θεωρείται κάθε περιοχή, οι C. Bettini, X.S. Wang και S. Jajodia στα [9],[10] έδωσαν ένα νέο ορισμό σύμφωνα με τον οποίο προσδιοριστές προστασίας θεωρούνται μόνο περιοχές τις οποίες επισκέπτεται συχνά ο χρήστης (LBQIDs). Αν ένας χρήστης εκτελεί συχνά μια διαδρομή, για να θεωρηθεί αυτή LBQID, θα πρέπει να εκτελείται τις ίδιες χρονικές στιγμές για έναν ελάχιστο αριθμό φορών. Για παράδειγμα, έστω ότι ένας χρήστης επισκέπτεται το νοσοκομείο τακτικά. Ένα LBQID θα μπορούσε να είναι το εξής:< <Σπίτι[15:00-16:00]>, <Νοσοκομείο[16:00-17:00]>, <Σπίτι[17:00-18:00]>> εάν συνέβαινε τουλάχιστον τρεις φορές σε μία εβδομάδα. Το βασικότερο μειονέκτημα του παραπάνω ορισμού είναι ότι δεν αναγνωρίζει σαν LBQID διαδρομές που επαναλαμβάνονται ανεξαρτήτως χρονικής στιγμής, όπως για παράδειγμα, μια διαδρομή που θα επαναλάμβανε ένας οδηγός ταξί σε μία μέρα ή μια μακρινή διαδρομή που θα επαναλάμβανε ένα πλοίο τέσσερις φορές μέσα σε ένα χρόνο.

2.3 Εξόρυξη συχνών διαδρομών

Μια πιο ρεαλιστική και πρωτοποριακή μέθοδος ανακάλυψης LBQIDs προτάθηκε από τους Α. Γκουλαλά-Διβάνη και Β. Βερύκιο στο [1]. Σύμφωνα με αυτή, τα LBQIDs ανακαλύπτονται αυτόματα από το σύστημα με εξόρυξη συχνών στοιχειοσυνόλων από τη ΒΔ στην οποία αποθηκεύονται οι πραγματικές διαδρομές (trajectories) που έχει ακολουθήσει κάθε χρήστης και παριστάνονται σε ένα 3-διάστατο επίπεδο (x, y, t), όπως φαίνεται στο Σχήμα 2.2.



Σχήμα 2.2 : Αναπαράσταση τροχιάς που αποτελεί LBQID για ένα χρήστη

Μια επίσης ρεαλιστική προσέγγιση έκαναν και οι G.Gidofalvi, Torben Bach Pedersen στο[7], οι οποίοι αναπαριστούν σε 3-διάστατο επίπεδο τα trajectories, αλλά για να βρουν τις συχνές διαδρομές που ακολουθεί ένας χρήστης, προβάλλουν στο επίπεδο (x, y) τα σημεία (x, y, t) από τα οποία πέρασε ο χρήστης και αναπαριστούν τις διαδρομές σαν ακολουθίες κελιών από τα οποία αυτός διέρχεται. Το μειονέκτημα και αυτής της τεχνικής είναι ότι βρίσκει συχνές διαδρομές που επαναλαμβάνονται σε ίδιες χρονικές στιγμές.

2.4 Εξόρυξη συχνών επεισοδίων

Στον τομέα της εξόρυξης συχνών ακολουθιών μέσω της εύρεσης συχνών επεισοδίων, έχουν προταθεί διάφοροι αλγόριθμοι σύμφωνα με το [12]. Οι πιο σημαντικοί από αυτούς είναι οι εξής:

Ο αλγόριθμος *WINEPI* [11]είναι ουσιαστικά ένα σύνολο από αλγορίθμους, που μπορούν να εφαρμοστούν σε οποιαδήποτε διατεταγμένη χρονικά ακολουθία

δεδομένων. Μπορεί να βρει επεισόδια που συμβαίνουν είτε σειριακά, με βάση μια προκαθορισμένη χρονική σειρά, είτε παράλληλα, χωρίς να υπάρχει δηλαδή κάποιος περιορισμός σε σχέση με το χρόνο. Από αυτά τα επεισόδια, μπορεί να βρει ποια είναι συχνά και ποια όχι. Αυτό το καταφέρνει διαπερνώντας τη ΒΔ με ένα παράθυρο ολίσθησης και βρίσκοντας το ποσοστό των παραθύρων στα οποία παρατηρείται το συγκεκριμένο επεισόδιο.

Ο αλγόριθμος *MINEPI* [11]είναι μια βελτίωση του παραπάνω αλγορίθμου για την εύρεση συχνών ακολουθιών, που βασίζεται στον ελάχιστο αριθμό φορών εμφάνισης κάθε ακολουθίας.

Ο αλγόριθμος *GSP* [13]σχεδιάστηκε για δεδομένα δοσοληψιών (transactions), όπου κάθε ακολουθία είναι μια ακολουθία από δοσοληψίες διατεταγμένες στο χρόνο. Ο αλγόριθμος αυτός καθορίζει τη μέγιστη χρονική διαφορά μεταξύ της πρώτης και της τελευταίας δοσοληψίας, καθώς επίσης και το μέγιστο και ελάχιστο κενό μεταξύ των στοιχείων της ακολουθίας.

Η παρούσα εργασία βασίστηκε στα [1], [2], [3], [7] και [13]. Οι χρήστες είναι αντικείμενα που κινούνται στους δρόμους ενός πραγματικού οδικού δικτύου μιας πόλης, που δημιουργήσαμε με χρήση ενός ειδικού προγράμματος παραγωγής κινουμένων αντικειμένων, το Generator του Brinkhoff. Τα αντικείμενα αυτά εισήχθησαν στο χωρικό δίκτυο μέσω της Oracle Spatial.

ΚΕΦΑΛΑΙΟ 3

3. Παραγωγή χώρο-χρονικών δεδομένων και κίνηση σε οδικό δίκτυο

Για την παραγωγή χώρο-χρονικών δεδομένων χρησιμοποιήθηκε ένα ειδικό πρόγραμμα παραγωγής χώρο-χρονικών δεδομένων, ο Generator του Thomas Brinkhoff. Το πρόγραμμα αυτό δέχεται σαν είσοδο δύο αρχεία, τα Oldenburg.node και Oldenburg.edge, με βάση τα οποία δημιουργεί το οδικό δίκτυο της πόλης Oldenburg. Έπειτα, με τη χρήση του αλγορίθμου GSTD(Generate SpatioTemporal Data) και κάποιων τυχαίων συναρτήσεων, παράγει αντικείμενα – χρήστες, που κινούνται επάνω στο οδικό δίκτυο. Το δίκτυο που δημιουργείται έχει τις ιδιότητες που χαρακτηρίζουν ένα πραγματικό οδικό δίκτυο και είναι οι εξής, όπως αναφέρονται στο [6]:

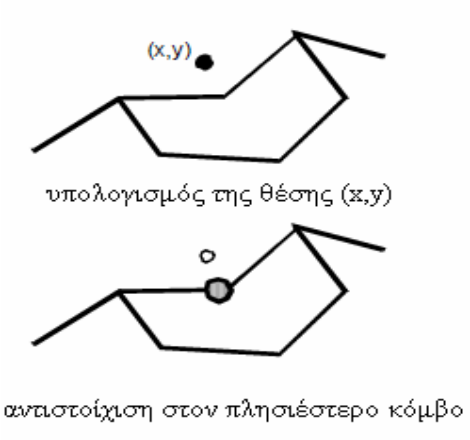
- Το οδικό δίκτυο αποτελείται από διασυνδέσεις, που επηρεάζουν την ταχύτητα των κινουμένων αντικειμένων.
- Η ταχύτητα των αντικειμένων μπορεί να επηρεαστεί αν ο αριθμός των αντικειμένων ξεπεράσει ένα κατώτατο όριο.
- Επειδή τα αντικείμενα επιλέγουν συνήθως την πιο σύντομη διαδρομή, για να φτάσουν στον προορισμό τους, αν η ταχύτητά τους αλλάξει κάποια στιγμή, λόγω κίνησης ή εξωτερικού παράγοντα, τότε μπορεί να αλλάξουν και διαδρομή.

Κάθε χώρο-χρονικό αντικείμενο obj_i χαρακτηρίζεται από τρεις παραμέτρους, το αναγνωριστικό του (obj.id), το χώρο ($obj.loc_i$) και το χρόνο ($obj.time_i$). Ο χώρος είναι

οι συντεταγμένες (x, y) του σημείου στο οποίο βρίσκεται το αντικείμενο χωρίς να είναι απαραίτητο ότι αντιστοιχούν σε κόμβους του οδικού δικτύου, που δημιουργήσαμε. Ο χρόνος στην πραγματικότητα είναι συνεχής. Εδώ όμως ο χρόνος είναι διακριτός. Αρχικά ορίζεται μια περίοδος T, που οριοθετείται από ένα κατώτερο χρονικό όριο t_{min} και ένα ανώτατο t_{max}. Η περίοδος T χωρίζεται σε n_t διαστήματα της μορφής $[t_i, t_{i+1})$ καθένα από τα οποία έχει τη χρονική σφραγίδα t_i, και ισχύει $t_0=t_{min}$, $t_{nt}=t_{max}$ και $t_{i+1}-t_i=\Delta t=(t_{max}-t_{min})/n_t$. Ένα αντικείμενο σταματά να υφίσταται είτε όταν φτάσει στον προορισμό του είτε όταν τελειώνει η περίοδος (δηλαδή το χρονικό διάστημα με τη μεγαλύτερη χρονοσφραγίδα).

3.1 Προσδιορισμός του κόμβου έναρξης

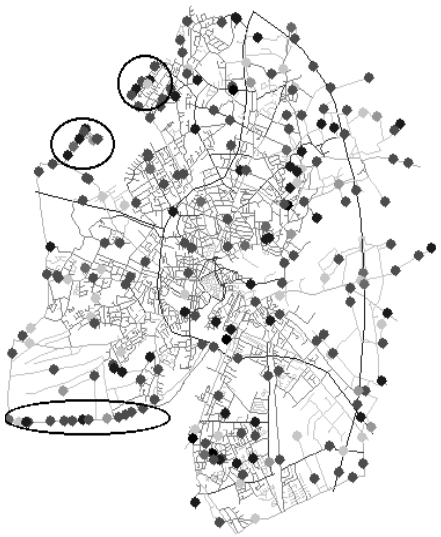
Ένα σημείο που θα παραχθεί για ένα κινούμενο αντικείμενο μπορεί να μην ανήκει σε κάποιο από τα σημεία του δικτύου που δημιουργήσαμε. Γι' αυτό θα πρέπει με την κατάλληλη διαδικασία να το ταυτίσουμε με κάποιο από τα σημεία του δικτύου. Πριν ξεκινήσει να κινείται ένα αντικείμενο θα πρέπει να βρεθεί ο κόμβος του δικτύου από τον οποίο θα ξεκινήσει. Υπάρχουν τρεις τρόποι, όπως περιγράφονται στο [6], και στηρίζονται στην ιδέα του «πλησιέστερου γείτονα» (Σχήμα 3.1), για να γίνει αυτή η διαδικασία:



Σχήμα 3.1 : Αντιστοίχιση στον πλησιέστερο γείτονα.

1. **Προσέγγιση που βασίζεται στο χώρο δεδομένων (DSO)**

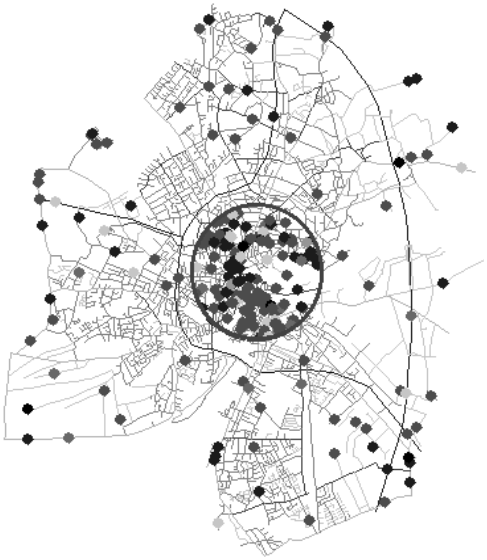
Με βάση την προσέγγιση αυτή, ο κόμβος έναρξης καθορίζεται από την πυκνότητα του δικτύου. Έτσι τα περισσότερα αντικείμενα έχουν σαν κόμβο έναρξης κάποιο κόμβο που βρίσκεται σε «αραιή» περιοχή του δικτύου. Στο Σχήμα 3.2 φαίνονται κυκλωμένες τέτοιες περιοχές.



Σχήμα 3.2 : Προσέγγιση που βασίζεται στο χώρο δεδομένων (DSO).

2. **Μέθοδος που βασίζεται στην περιοχή (RB).**

Αποτελεί βελτίωση της παραπάνω τεχνικής. Εισάγονται πλέον κάποια στατιστικά στοιχεία και η έννοια της περιοχής. Κάθε περιοχή χαρακτηρίζεται από την πιθανότητα να ανήκει σε αυτή ο κόμβος έναρξης του αντικειμένου.



Σχήμα 3.3 : Μέθοδος που βασίζεται στην περιοχή.

3. **Μέθοδος που βασίζεται στο δίκτυο (NB).**

Η τελευταία τεχνική βασίζεται σε μια συνάρτηση κατανομής. Αν επιλέξουμε την ομοιόμορφη κατανομή, τότε κάθε κόμβος έχει ίσες πιθανότητες με τους υπόλοιπους να είναι κόμβος έναρξης.

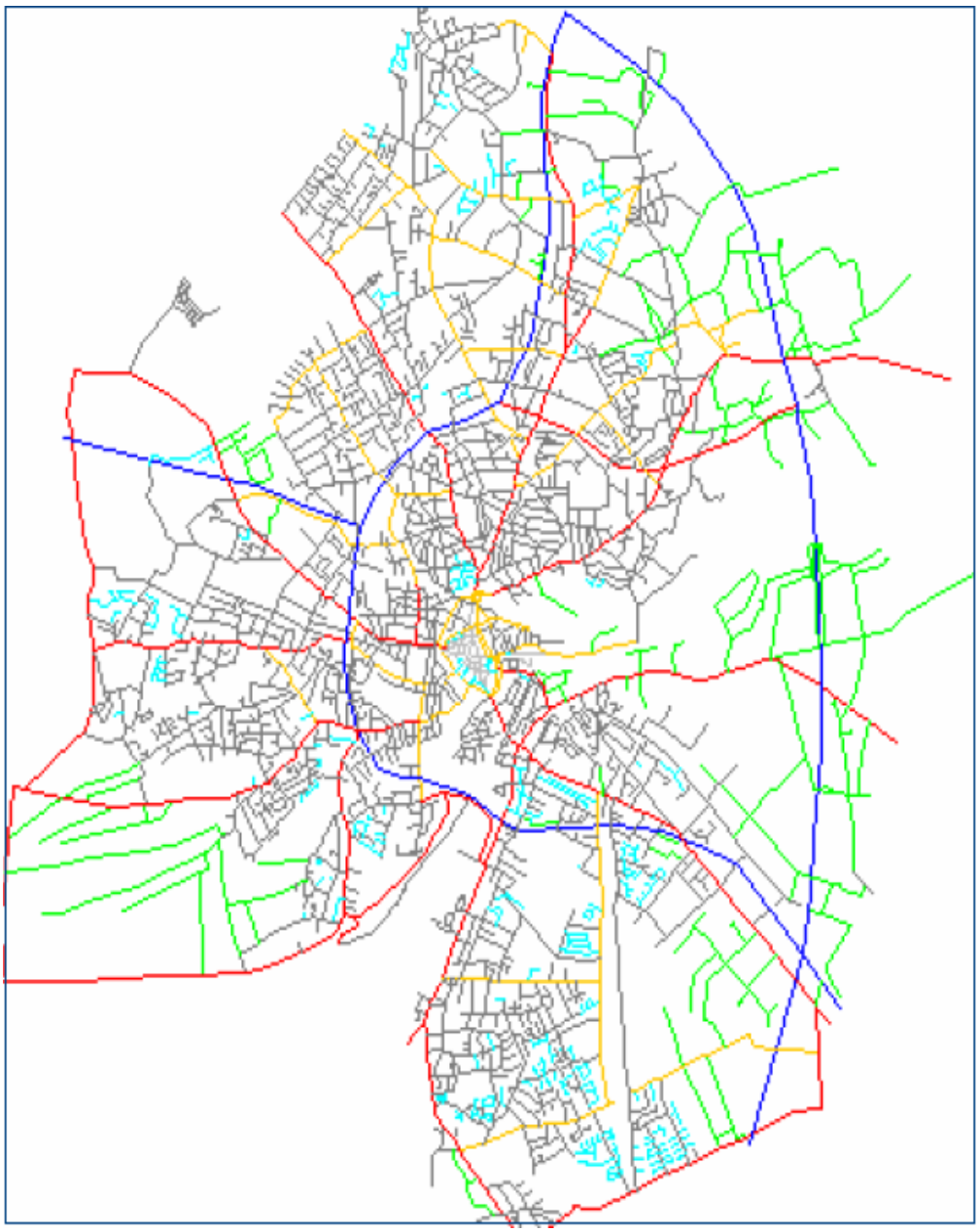
Σχήμα 3.4 : Μέθοδος που βασίζεται στο δίκτυο.

3.2 Η κίνηση των αντικειμένων

Η συνάρτηση που παράγει τα κινούμενα αντικείμενα, εξαρτάται από το χρόνο. Κάθε αντικείμενο κινείται μεταξύ δύο διαδοχικών χρονοσφραγίδων (t_i, t_{i+1}) από τη θέση loc_i που είχε τη χρονική στιγμή ti, στη θέση loc_{i+1}, σύμφωνα με τη διαδρομή που έχει υπολογιστεί. Η διαδρομή αυτή υπολογίζεται με κάποιο αλγόριθμο εύρεσης συντομότερου μονοπατιού (όπως π.χ. ο Dijkstra) από την εκκίνηση του αντικειμένου και δίνει σαν αποτέλεσμα το συντομότερο δρόμο από τον κόμβο έναρξης στον κόμβο προορισμού. Είναι προκαθορισμένη και δεν αλλάζει κατά την κίνηση του αντικειμένου. Η ταχύτητα του αντικειμένου θεωρείται σταθερή. Αυτό είναι μια μη ρεαλιστική παραδοχή, αφού η ταχύτητα μεταβάλλεται και με βάση αυτή τη μεταβολή μπορεί να αλλάξει και η διαδρομή που αρχικά είχε υπολογιστεί σαν συντομότερη.

3.3 Κατασκευή δικτύου στην Oracle

Όπως αναφέραμε προηγουμένως, τα κινούμενα (χώρο - χρονικά) αντικείμενα πρέπει να τα εισάγουμε σε ένα οδικό δίκτυο. Για το σκοπό αυτό κατασκευάζουμε το δίκτυο της πόλης Oldenburg, που φαίνεται στο Σχήμα 3.5.



Σχήμα 3.5: Δίκτυο Oldenburg.

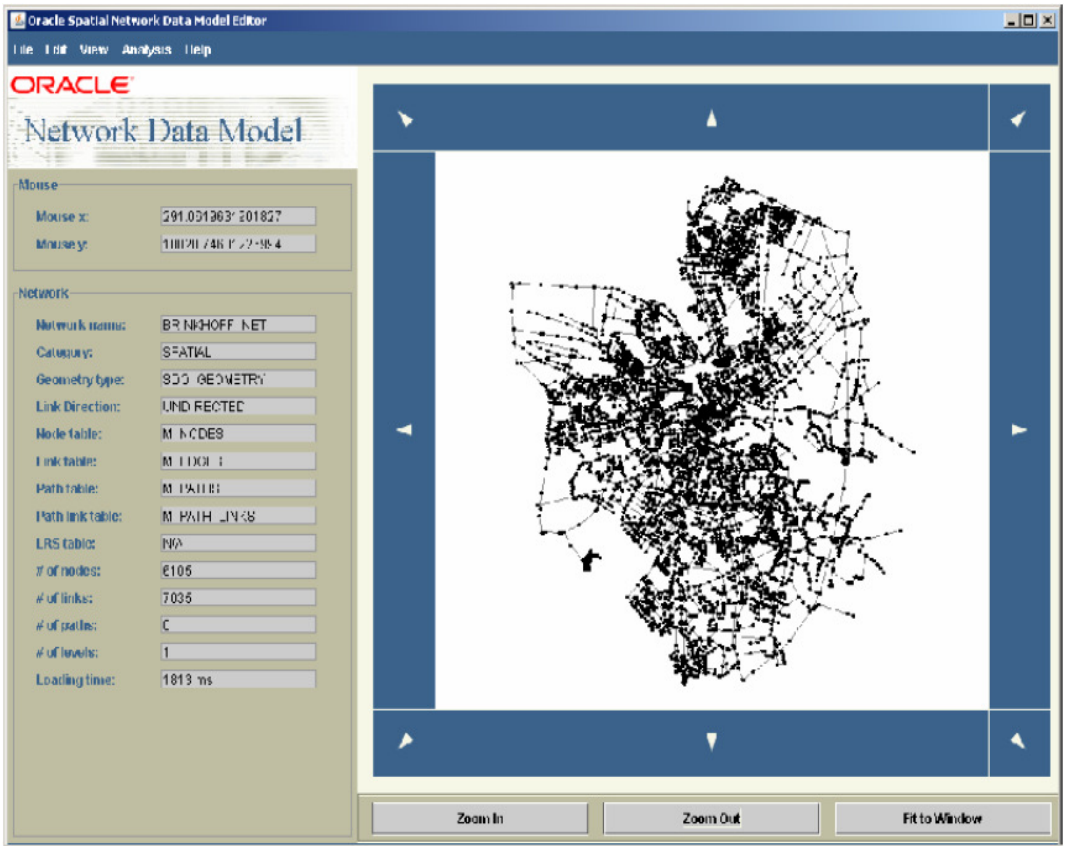
Ο Δ. Δούρας στο [6], για την κατασκευή του χωρικού δικτύου και την επικοινωνία του run time environment της Java με το ΣΔΒΔ της Oracle χρησιμοποίησε ένα java script το οποίο χρησιμοποιεί το πρωτόκολλο JDBC. Για τη διαχείριση των χωρικών δεδομένων του δικτύου μέσω της java ενσωματώθηκε το Network Data Model Java Interface.

Το παραπάνω δίκτυο υλοποιείται ουσιαστικά με δύο κωδικοποιημένα αρχεία (.txt). Το ένα (Oldenburg.node) περιέχει πληροφορίες για τους κόμβους του δικτύου, όπως το αναγνωριστικό κάθε κόμβου (NODE_ID) και τη γεωμετρία του κόμβου σε συντεταγμένες (x,y), και το άλλο (Oldenburg.edge) περιέχει πληροφορίες για τις ακμές, όπως το αναγνωριστικό κάθε ακμής (LINK_ID) και τους εναρκτήριους και καταληκτικούς κόμβους κάθε ακμής.

Για την κατασκευή του χωρικού δικτύου και την αποθήκευσή του στη Β.Δ. της Oracle, ακολουθήσαμε τα παρακάτω βήματα:

1. Δημιουργήσαμε ένα πίνακα κόμβων και ένα πίνακα ακμών με χρήση SQL στην Oracle.
2. Χρησιμοποιήσαμε τα αρχεία .node και .edge ως αρχεία εισόδου για ανάγνωση και αποθήκευση στη Java.
3. Μετατρέψαμε τα αρχεία αυτά σε δεδομένα που μπορούν να εισαχθούν στους πίνακες της Β.Δ. της Oracle. Συγκεκριμένα, με τη δομή JGeometry του Network Data Model Java Interface μετατρέψαμε τις συντεταγμένες των σημείων σε γεωμετρίες της Oracle Spatial.
4. Εισήγαμε με SQL τα μετασχηματισμένα δεδομένα στους πίνακες κόμβων και ακμών στην Oracle, ενημερώσαμε τα μεταδεδομένα για τους πίνακες αυτούς και κατασκευάσαμε τα απαραίτητα χωρικά ευρετήρια.

Αφού κατασκευάσαμε το δίκτυο, μπορούμε να το ελέγξουμε και να το διαχειριστούμε, μέσα από τα ειδικό εργαλείο που παρέχει η Oracle, το Network Editor, το οποίο φαίνεται στο Σχήμα 3.6.

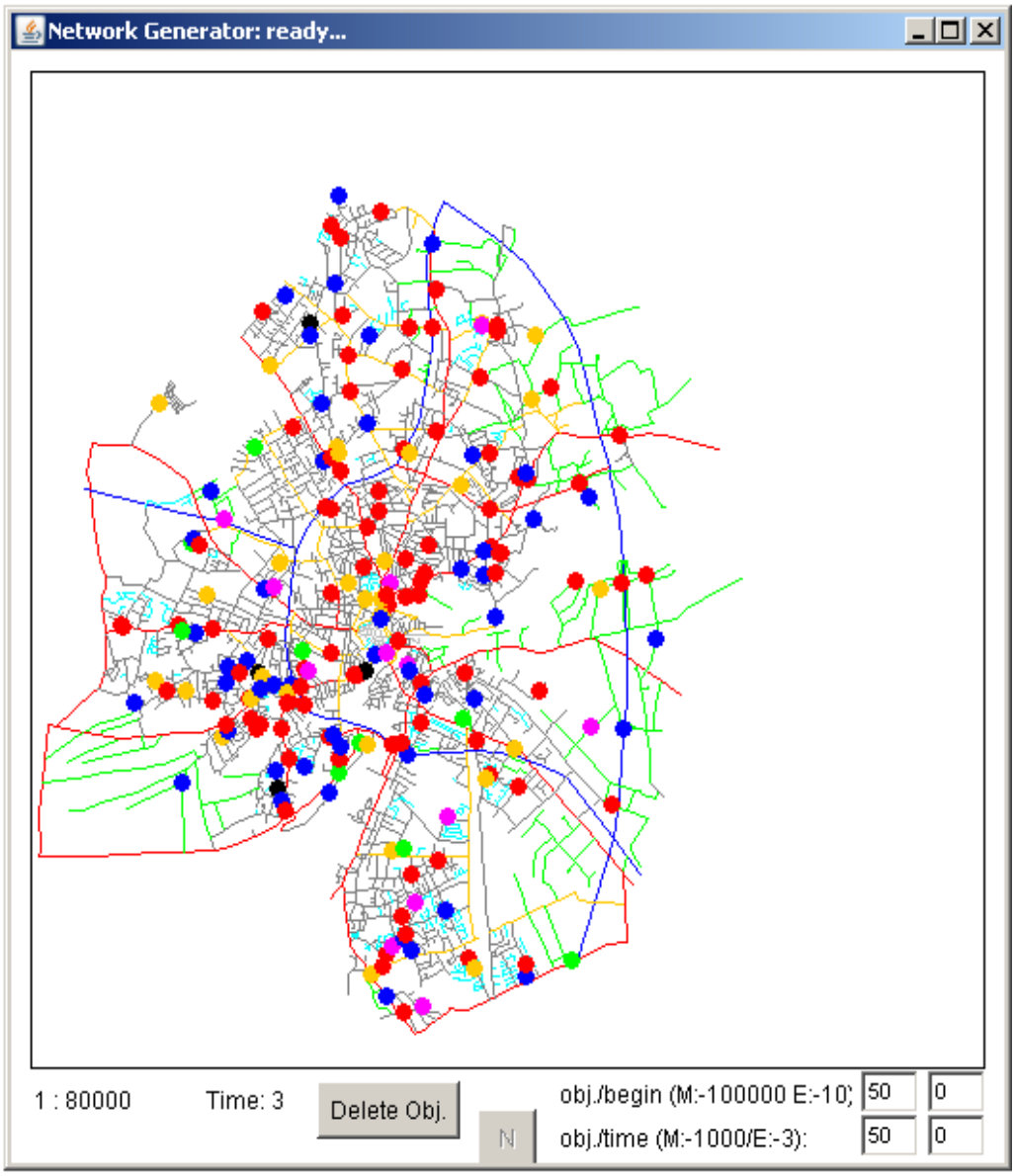


Σχήμα 3.6: Δίκτυο Oldenburg - Oracle Network Editor

3.4 Εισαγωγή των χώρο-χρονικών αντικειμένων στο δίκτυο της Oracle

Αφού κατασκευάσουμε το δίκτυο, μπορούμε να εισάγουμε σε αυτό τα χώρο - χρονικά αντικείμενα, που έχουμε παράγει με το Generator του Brinkhoff, όπως περιγράψαμε προηγουμένως. Εκτός από τα αρχεία .node και .edge που αποθηκεύσαμε στη Β.Δ της Oracle, καθορίζουμε επίσης των αριθμό των χρηστών που κινούνται στο δίκτυο και

μια περίοδο T κατά την οποία θα γίνονται αυτές οι κινήσεις. Έπειτα ο Generator παράγει τα αντικείμενα, όπως φαίνεται στο Σχήμα 3.7.



Σχήμα 3.7: Κινούμενα αντικείμενα στο Oldenburg.

Ο Generator δίνει τελικά σαν έξοδο ένα αρχείο κειμένου (OldenburgGen.dat) το οποίο περιλαμβάνει τις διαδρομές που ακολούθησαν οι χρήστες, καταγράφοντας το αναγνωριστικό κάθε χρήστη (USER_ID), το σημείο (x,y) πάνω στο δίκτυο, στο οποίο βρίσκεται και την χρονική στιγμή (t) κατά την οποία βρίσκεται σε αυτό το σημείο.

Για την εισαγωγή αυτών των δεδομένων στην Oracle, χρησιμοποιήθηκε ο κώδικας σε Java (createPHL.java) και ακολουθήθηκαν τα παρακάτω βήματα:

1. Δημιουργήσαμε ένα πίνακα που αποθηκεύει τις ενημερώσεις θέσης των χρηστών.
2. Χρησιμοποιήσαμε το αρχείο OldenburgGen.dat ως αρχείο εισόδου για ανάγνωση και αποθήκευση στη Java.
3. Μετατρέψαμε το αρχεία αυτό σε δεδομένα που μπορούν να εισαχθούν στους πίνακες της Β.Δ. της Oracle. Συγκεκριμένα, με τη δομή JGeometry του Network Data Model Java Interface μετατρέψαμε τις συντεταγμένες των σημείων σε γεωμετρίες της Oracle Spatial.
4. Εισήγαμε με SQL τα νέα δεδομένα στον πίνακα των διαδρομών στην Oracle, ενημερώσαμε τα μεταδεδομένα για τον πίνακα αυτόν και κατασκευάσαμε το απαραίτητο χωρικό ευρετήριο.

Τελικά στη ΒΔ έχουμε αποθηκευμένο πλέον τον πίνακα που περιέχει ολόκληρο το ιστορικό των κινήσεων όλων των χρηστών. Σε αυτόν τον πίνακα θα βασιστούμε για να εφαρμόσουμε τον αλγόριθμο ανακάλυψης συχνών διαδρομών, που ακολουθεί κάθε χρήστης.

3.5 Μειονέκτημα του Generator του Brinkhoff

Το μειονέκτημα του Generator του Brinkhoff είναι ότι τα κινούμενα αντικείμενα κατά τη διάρκεια της Τ δε διέρχονται ποτέ από το ίδιο σημείο παραπάνω από μια φορά. Αυτό φυσικά είναι μη ρεαλιστικό. Άμεση συνέπεια αυτού του γεγονότος είναι ότι δεν

μπορούμε να βρούμε διαδρομές του ίδιου αντικειμένου που να επαναλαμβάνονται. Για να διορθώσουμε αυτό το μειονέκτημα και να δημιουργήσουμε ένα πιο ρεαλιστικό μοντέλο, θεωρούμε ότι οι 10 διαδρομές που κάνουν 10 χρήστες σε μια περίοδο T, δεν είναι 10 διαφορετικές διαδρομές 10 χρηστών, αλλά 10 διαφορετικές διαδρομές του ίδιου χρήστη σε 10T. Για παράδειγμα, αν ο Generator δώσει σαν έξοδο ένα αρχείο με τις διαδρομές που έκαναν 10 χρήστες σε μια περίοδο T = 20, τότε θα ισχύουν τα εξής:

- Τα ids των χρηστών θα είναι 0 έως 9.
- Τα σημεία (x, y, t) θα είναι το πολύ 20 για κάθε χρήστη, όσες είναι δηλαδή και οι διακριτές χρονικές στιγμές της περιόδου T.
- Κάθε χρήστης θα διέρχεται από διαφορετικά σημεία (x, y).

Στο Σχήμα 3.8 παρουσιάζεται ένας πίνακας για τους πρώτους 5 χρήστες.

USER_ID	X	Y	T
0	3524	19989	0
0	3265.28485	19977.3669	1
0	3257	19910	2
1	14498	20720	0
1	14328.0366	20960.1922	1
1	13705.3291	20707.6448	2
1	13105.882	20414.8305	3
1	12619.9014	19980.9492	4
1	12276.0324	19434.7661	5
1	12051.1693	18933.0301	6
1	12177.3114	18556.3878	7
1	12216.3621	18118.6982	8
1	11941.4319	17832.1611	9
1	11727.7391	17439.3875	10
1	11743.1193	17112.3083	11
1	11988.9631	16784.6689	12
1	12583.3015	16631.0355	13
1	12865.6319	16737.9687	14
1	12995	16785	15
2	16305	14696	0
2	16133.5261	14453.1509	1
2	15892.0224	14847.0332	2
2	15537.2261	15235.896	3
2	15705.6997	15465.002	4
2	15727.981	15760.2808	5
2	15609.973	16033.6072	6
2	15492.8859	16300.8736	7
2	15547	16343	8
3	10954	6921	0
3	10773.8346	6923.40474	1
3	10598.7876	7105.03243	2
3	10766.9094	7284.06725	3
3	10944.6809	7463.02916	4
3	11120.3763	7644.00175	5
3	11293.5604	7827.40657	6
3	11225.6805	8009.30539	7
3	11050.2184	8190.53212	8
3	10874.7563	8371.75885	9
3	10724.0591	8573.80905	10
4	13604	11564	0
4	13624.4339	11976.9581	1
4	13510.9472	12436.4653	2
4	13792.8137	12533.188	3
4	14074.6841	12629.8992	4
4	14356.5546	12726.6105	5
4	14639.6289	12838.732	6
4	14940.4055	13170.7505	7
4	14924	13318	8
5	15365	12631	0
5	15221.2159	12370.5577	1
5	14912.7806	12520.2405	2
5	14770.8934	12258.1886	3
5	14626.74	11997.3749	4
5	14481.5945	11737.1132	5
5	14329.7589	11482.6999	6
5	14096.5936	11297.1253	7
5	13858	11118.8887	8
5	13863.8801	10614.4361	9
5	13906.3167	10112.9347	10
5	13984.0554	9614.46011	11
5	14043.9215	9113.5337	12
5	14055.6798	8610.21404	13
5	13859.2819	8269.1355	14
5	13411.8584	8246.61259	15
5	12990.8619	8220.81849	16
5	12543.6363	8202.10725	17
5	12322.9317	8178.74222	18
5	12316.6857	7880.80768	19
5	12311.0766	7582.86146	20

Σχήμα 3.8: Πίνακας διαδρομών των χρηστών

Στην παρούσα εργασία, για να διορθώσουμε αυτό το μειονέκτημα, επεξεργαστήκαμε το αρχείο εξόδου του Generator, που περιέχει τις διαδρομές των χρηστών με την παραπάνω μορφή, ως εξής:

- Αλλάζουμε τα ids των χρηστών, έτσι ώστε ανά 10 να αναφέρονται στον ίδιο χρήστη.
- Θεωρούμε ότι οι 10 διαδρομές, δε γίνονται πλέον σε 1T ταυτόχρονα, αλλά σε 10T.

Για να το κάνουμε αυτό, δημιουργήσαμε ένα πρόγραμμα με γλώσσα προγραμματισμού C, με το οποίο επεξεργαζόμαστε το αρχείο Oldenburg.dat, το οποίο περιέχει όλες τις διαδρομές των χρηστών με χρονολογική σειρά. Το πρόγραμμά μας δημιουργεί ένα καινούριο αρχείο .txt διαβάζοντας το Oldenburg.dat, το οποίο περιέχει τις διαδρομές κάθε χρήστη και πάλι με χρονολογική σειρά.

Εάν θέλαμε να παρομοιάσουμε το παραπάνω παράδειγμα με ένα παράδειγμα από την καθημερινότητα, θα μπορούσαμε να πούμε ότι για κάθε χρήστη που χρησιμοποιεί μια συσκευή GPS και ζητά την παροχή μιας LBS υπηρεσίας, αποθηκεύονται στη ΒΔ οι διαδρομές που ακολούθησε. Ο χρήστης με id=0, παραδείγματος χάριν, έκανε 10 διαδρομές σε 10T. Το τελευταίο σημείο κάθε διαδρομής του δεν είναι απαραίτητο να είναι το πρώτο της αμέσως επόμενης, γιατί κάνουμε τη ρεαλιστική παραδοχή ότι μπορεί ο χρήστης, να έκλεισε το GPS του όταν τελείωσε τη διαδρομή και δεν το άνοιξε αμέσως μόλις ξεκίνησε την επόμενη διαδρομή του, αλλά σε κάποιο σημείο αυτής. Στο Σχήμα 3.9 φαίνεται πώς μετασχηματίζεται ο παραπάνω πίνακας. Πλέον παριστάνει τις 5 πρώτες διαδοχικές διαδρομές του χρήστη με id=0.

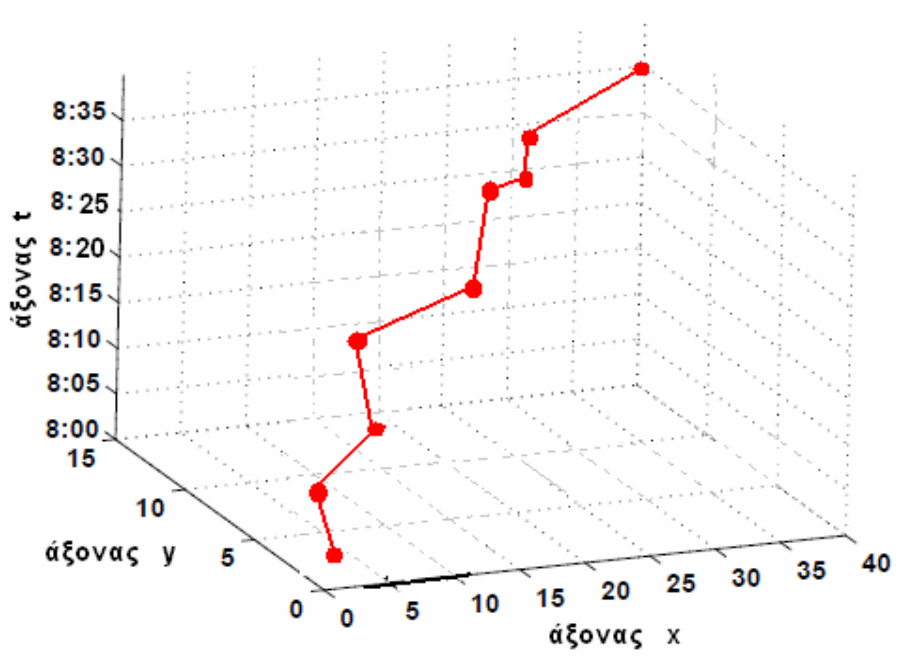
USER_ID	X	Y	T
0	3524	19989	0
0	3265.28485	19977.3669	1
0	3257	19910	2
0	14498	20720	0
0	14328.0366	20960.1922	1
0	13705.3291	20707.6448	2
0	13105.882	20414.8305	3
0	12619.9014	19980.9492	4
0	12276.0324	19434.7661	5
0	12051.1693	18933.0301	6
0	12177.3114	18556.3878	7
0	12216.3621	18118.6982	8
0	11941.4319	17832.1611	9
0	11727.7391	17439.3875	10
0	11743.1193	17112.3083	11
0	11988.9631	16784.6689	12
0	12583.3015	16631.0355	13
0	12865.6319	16737.9687	14
0	12995	16785	15
0	16305	14696	0
0	16133.5261	14453.1509	1
0	15892.0224	14847.0332	2
0	15537.2261	15235.896	3
0	15705.6997	15465.002	4
0	15727.981	15760.2808	5
0	15609.973	16033.6072	6
0	15492.8859	16300.8736	7
0	15547	16343	8
0	10954	6921	0
0	10773.8346	6923.40474	1
0	10598.7876	7105.03243	2
0	10766.9094	7284.06725	3
0	10944.6809	7463.02916	4
0	11120.3763	7644.00175	5
0	11293.5604	7827.40657	6
0	11225.6805	8009.30539	7
0	11050.2184	8190.53212	8
0	10874.7563	8371.75885	9
0	10724.0591	8573.80905	10
0	13604	11564	0
0	13624.4339	11976.9581	1
0	13510.9472	12436.4653	2
0	13792.8137	12533.188	3
0	14074.6841	12629.8992	4
0	14356.5546	12726.6105	5
0	14639.6289	12838.732	6
0	14940.4055	13170.7505	7
0	14924	13318	8
0	15365	12631	0
0	15221.2159	12370.5577	1
0	14912.7806	12520.2405	2
0	14770.8934	12258.1886	3
0	14626.74	11997.3749	4
0	14481.5945	11737.1132	5
0	14329.7589	11482.6999	6
0	14096.5936	11297.1253	7
0	13858	11118.8887	8
0	13863.8801	10614.4361	9
0	13906.3167	10112.9347	10
0	13984.0554	9614.46011	11
0	14043.9215	9113.5337	12
0	14055.6798	8610.21404	13
0	13859.2819	8269.1355	14
0	13411.8584	8246.61259	15
0	12990.8619	8220.81849	16
0	12543.6363	8202.10725	17
0	12322.9317	8178.74222	18
0	12316.6857	7880.80768	19
0	12311.0766	7582.86146	20

Σχήμα 3.9: Πίνακας διαδρομών του χρήστη με id=0

ΚΕΦΑΛΑΙΟ 4

4. Εξόρυξη γνώσης από χώρο – χρονικά δεδομένα

Οι διαδρομές (trajectories) που ακολουθούν οι χρήστες – κινούμενα αντικείμενα είναι ουσιαστικά ακολουθίες τοποθεσιών, απ' τις οποίες διέρχεται ο κάθε χρήστης. Ένα trajectory είναι δηλαδή μια ακολουθία από σημεία (x, y, t), που δείχνουν σε ποιο σημείο (x, y) βρίσκεται ο χρήστης τη χρονική στιγμή t. Το Σχήμα 4.1 απεικονίζει τη διαδρομή (trajectory) ενός χρήστη που είναι η ακολουθία <(3,2,8:00), (8,7,8:05), (10,6,8:10), (16,14,8:11), (25,14,8:20), (26,13,8:26), (30,15,8:30), (31,14,8:35), (40,15,8:38)>

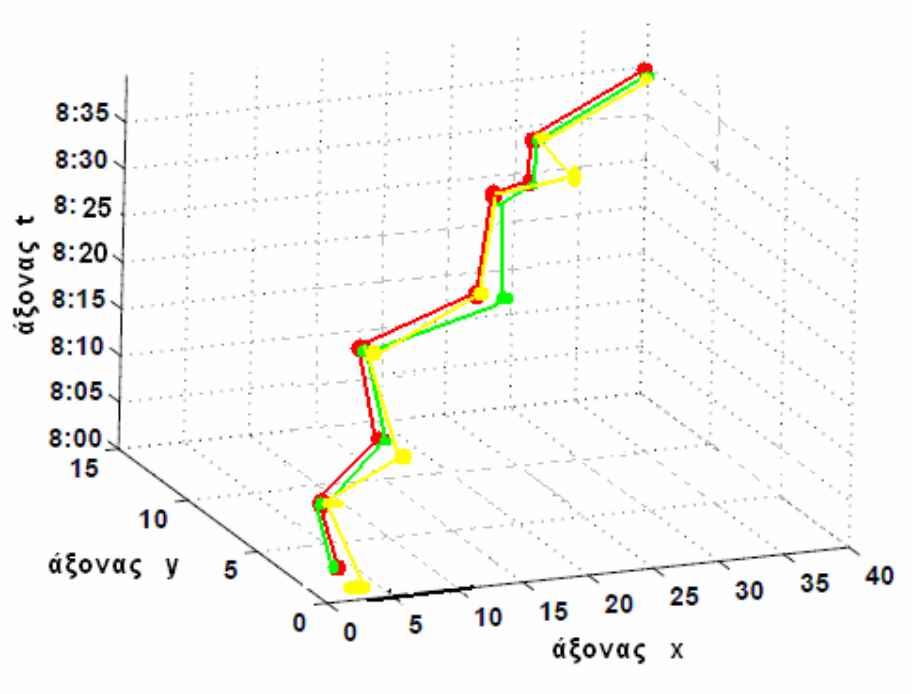


Σχήμα 4.1: Trajectory χρήστη

Πολλές φορές όμως οι συσκευές που παρέχουν LBS υπηρεσίες στέλνουν μη ακριβείς μετρήσεις που παρουσιάζουν κάποιο θόρυβο στα δεδομένα με αποτέλεσμα η

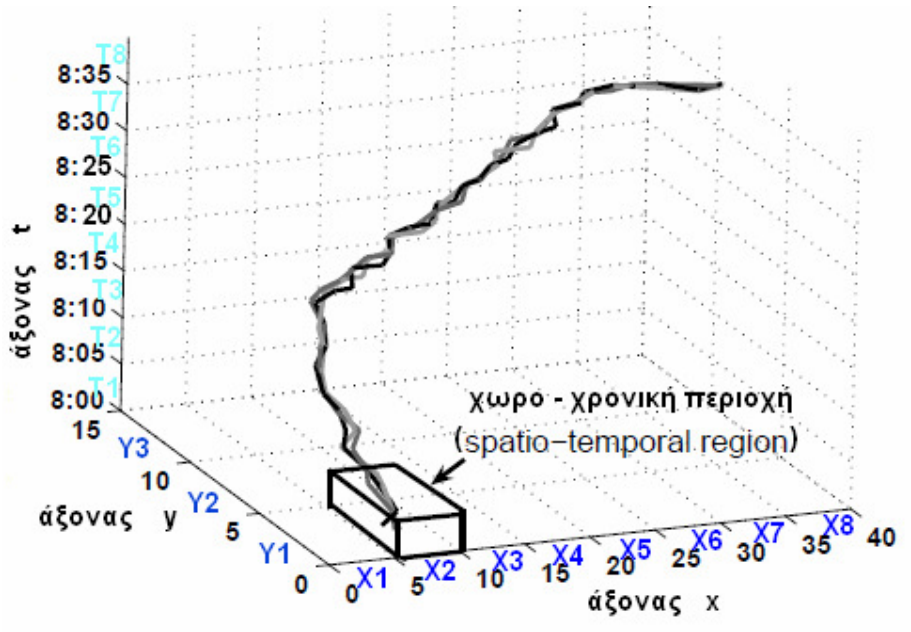
διαδρομή να παρεκκλίνει λίγο από την πραγματική. Αυτή η παρέκκλιση μπορεί να οφείλεται σε μια ολιγόλεπτη καθυστέρηση μετάδοσης των δεδομένων ή στη χρήση κάποιου διαφορετικού δρόμου στη διαδρομή. Για παράδειγμα στο Σχήμα 4.2 παριστάνονται τρεις διαδρομές, που αν και είναι όμοιες κατά το μεγαλύτερο μέρος τους, θεωρούνται διαφορετικές. Οι διαδρομές αυτές είναι οι εξής ακολουθίες:

- <(3,2,8:00), (8,7,8:05), (10,6,8:10), (16,14,8:11), (25,14,8:20), (26,13,8:26), (30,15,8:30), (31,14,8:35), (40,15,8:38)>
- <(3,2,8:00), (8,7,8:05), (10,6,8:10), (16,14,8:11), (26,14,8:21), (26,13,8:26), (30,15,8:30), (31,14,8:35), (40,15,8:38)>
- <(4,1,8:00), (8,7,8:05), (11,7,8:10), (16,14,8:11), (25,14,8:20), (26,13,8:26), (31,15,8:31), (31,14,8:35), (40,15,8:38)>



Σχήμα 4.2: Τρεις διαδρομές(trajectories) του ίδιου χρήστη

Για να μη θεωρούνται διαφορετικές τέτοιου είδους διαδρομές, θα πρέπει τα trajectories να αποτελούνται από πιο γενικευμένες περιοχές, έτσι ώστε αν μοιάζουν αρκετά μεταξύ τους τότε να θεωρούνται ίδια. Μια γενικευμένη περιοχή κατασκευάζεται σύμφωνα με το [7] αν αντικαταστήσουμε τα σημεία (x, y, t) με χώρο – χρονικές περιοχές. Για να το κάνουμε αυτό, χωρίζουμε τους άξονες x, y, t σε διαστήματα. Τα διαστήματα κάθε άξονα θα πρέπει να είναι ίσα μεταξύ τους. Κάθε χώρο – χρονική περιοχή χαρακτηρίζεται από μια ετικέτα XiYjTk, όπου Xi το διάστημα του άξονα x, Yj το διάστημα του άξονα y και Tk το διάστημα του άξονα t, που περιέχουν αντίστοιχα τα x, y, t. Το Σχήμα 4.3 απεικονίζει μια χώρο – χρονική περιοχή.



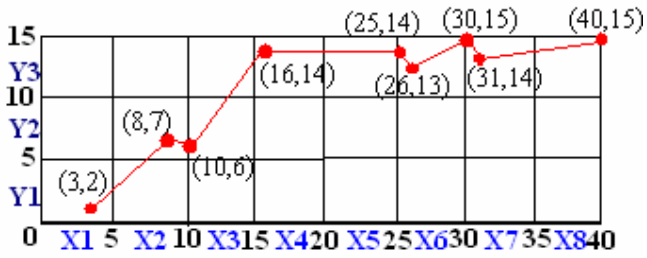
Σχήμα 4.3: Χώρο – χρονική περιοχή

Με βάση το παραπάνω σχήμα, οι διαδρομές που παριστάνονται στο σχήμα 4.2 είναι πλέον οι εξής:

- <X1Y1T1, X2Y2T2, X3Y2T3, X4Y3T3, X6Y3T5, X6Y3T6, X7Y4T7, X7Y3T8, X9Y4T8>
- <X1Y1T1, X2Y2T2, X3Y2T3, X4Y3T3, X6Y3T5, X6Y3T6, X7Y4T7, X7Y3T8, X9Y4T8>
- <X1Y1T1, X2Y2T2, X3Y2T3, X4Y3T3, X6Y3T5, X6Y3T6, X7Y4T7, X7Y3T8, X9Y4T8>

Παρατηρούμε ότι πλέον οι διαδρομές είναι ακριβώς ίδιες, το οποίο ήταν και το ζητούμενο.

Για να δούμε από ποια τετράγωνα (κελιά) του επιπέδου, που σχηματίζουν οι άξονες (x, y), διέρχεται ο χρήστης, ανεξαρτήτως χρόνου, προβάλλουμε τα σημεία (x, y, t) στο επίπεδο (x, y), όπως φαίνεται στο Σχήμα 4.5, όπου προβάλλουμε τη διαδρομή του Σχήματος 4.1.



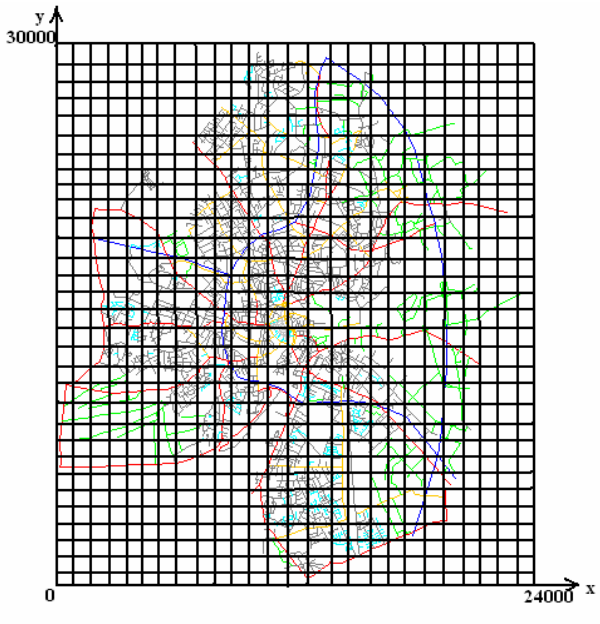
Σχήμα 4.5: Προβολή σημείων (x,y,t) στο επίπεδο (x,y).

Παρατηρούμε ότι στο επίπεδο (x,y) ο παραπάνω χρήστης διέρχεται από τα τετράγωνα (κελιά): <X1Y1, X2Y2, X3Y2, X4Y3, X6Y3, X6Y3, X7Y4, X7Y3, X9Y4>. Εναλλακτικά σύμφωνα με το [1] μπορούμε να αντικαταστήσουμε τα XiYj με την ετικέτα $C_{i,j}$, όπου i το διάστημα του άξονα x και j το διάστημα του άξονα y, στο οποίο ανήκουν οι συντεταγμένες του σημείου (x, y) αντίστοιχα. Οπότε η παραπάνω

διαδρομή γράφεται πλέον σαν ακολουθία κελιών ως εξής: < $C_{1,1}$, $C_{2,2}$, $C_{3,2}$, $C_{4,3}$, $C_{6,3}$, $C_{6,3}$, $C_{7,4}$, $C_{7,3}$, $C_{9,4}$ >.

Στην παρούσα εργασία εφαρμόζουμε την παραπάνω μέθοδο ως εξής: τα κινούμενα αντικείμενα διέρχονται από τους κόμβους του δικτύου που δημιουργήσαμε με την Oracle. Στον άξονα x τα σημεία ξεκινούν από το 0 και φτάνουν έως 24.000 και στον άξονα y από 0 έως 30.000. Μπορούμε λοιπόν να χωρίσουμε τους δύο άξονες σε διαστήματα ίσου μήκους και να προβάλλουμε τα χώρο – χρονικά σημεία (x, y, t) στο επίπεδο (x,y). Το Σχήμα 3.5 χωρίζεται πλέον σε κελιά και παίρνουμε σαν αποτέλεσμα το Σχήμα 4.6.

Για να το κάνουμε αυτό, δημιουργήσαμε ένα πρόγραμμα με τη C, με το οποίο διαβάζουμε το αρχείο .txt, που περιέχει όλα τα χώρο – χρονικά σημεία (x, y, t) από τα οποία διέρχεται κάθε χρήστης διατεταγμένα με χρονολογική σειρά, και αντιστοιχίζουμε τα χωρικά σημεία (x, y) σε κελιά $C_{i,j}$.



Σχήμα 4.6: Προβολή σημείων (x,y,t) στο επίπεδο (x,y) για το δίκτυο Oldenburg.

ΚΕΦΑΛΑΙΟ 5

5. Ανακάλυψη συχνών διαδρομών

Πολλές φορές η συμπεριφορά ή οι ενέργειες των χρηστών μπορούν να περιγραφούν από ακολουθίες γεγονότων. Ένα επεισόδιο ορίζεται σαν μια συλλογή από γεγονότα που συμβαίνουν το ένα κοντά στο άλλο σε μια διατεταγμένη χρονικά ακολουθία. Για το πρόβλημα της ανακάλυψης συχνών επεισοδίων έχουν προταθεί διάφοροι αλγόριθμοι. Στην παρούσα εργασία τα γεγονότα θα είναι τα κελιά και τα επεισόδια οι μεταβάσεις - διαδρομές από το ένα κελί στο άλλο. Εμείς θα αναλύσουμε και θα προσαρμόσουμε στις ανάγκες του προβλήματος εύρεσης των συχνών διαδρομών, που ακολουθεί ο χρήστης, τον αλγόριθμο WINEPI, που περιγράφεται αναλυτικά στο [11].

5.1 Ακολουθίες κελιών και μεταβάσεις

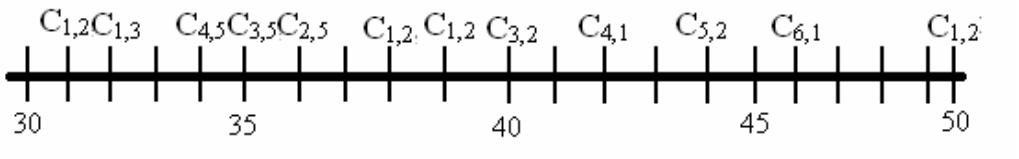
Σκοπός αυτού του κεφαλαίου είναι να περιγραφεί ο αλγόριθμος με τον οποίο θα αναλυθούν οι ακολουθίες των κελιών από τα οποία διέρχεται ο χρήστης έτσι ώστε να ανακαλυφθούν συχνές μεταβάσεις από το ένα κελί στο άλλο (frequent subtrajectories). Πρώτα θα παρουσιάσουμε την ιδέα των ακολουθιών κελιών και έπειτα τις μεταβάσεις (subtrajectories).

5.1.1 Ακολουθίες κελιών

Ο αλγόριθμος εύρεσης συχνών διαδρομών δέχεται σαν είσοδο μια ακολουθία από κελιά. Δεδομένου ενός συνόλου Ε από «ετικέτες» κελιών ($C_{i,j}$), μια ακολουθία S

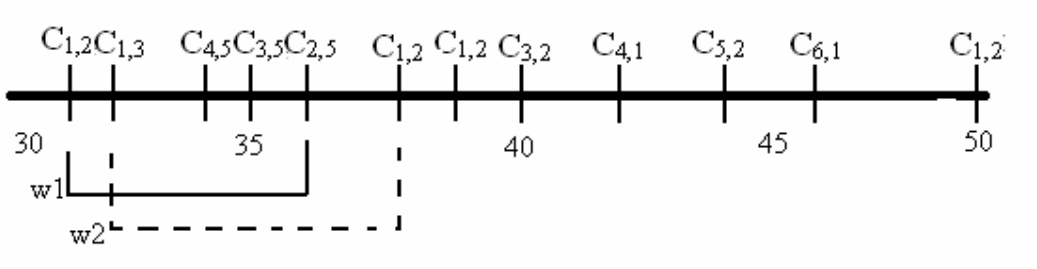
πάνω στο E είναι μια τριπλέτα (s, T_s, T_e), όπου s είναι μία διατεταγμένη στο χρόνο ακολουθία κελιών, δηλαδή $s = <A_1,A_2,\ldots\ldots,A_n>$, $A_i \in E$ για i=1,…..n, T_s είναι ο χρόνος έναρξης της ακολουθίας και T_e ο χρόνος που τελειώνει η ακολουθία.

Το Σχήμα 5.1 δείχνει την ακολουθία (s, 31, 51), όπου $s = <C_{1,2}, C_{1,3}, C_{4,5}, C_{3,5}, C_{2,5}, C_{1,2}, C_{1,2}, C_{3,2}, C_{4,1}, C_{5,2}, C_{6,1}, C_{1,2}>$. Ο χρήστης διέρχεται από τα παραπάνω κελιά, με αυτή τη χρονολογική σειρά, από τη στιγμή 29 έως την 50, δηλαδή στο χρονικό διάστημα [29,51).



Σχήμα 5.1: Ακολουθία κελιών

Ορίζουμε ένα *παράθυρο*(window) σαν ένα μέρος της ακολουθίας κελιών, με συγκεκριμένο μέγεθος – πλάτος, μέσα στο οποίο θα πρέπει να συμβαίνει μια μετάβαση. Έπειτα θεωρούμε την ακολουθία κελιών σαν μια ακολουθία από μερικώς επικαλυπτόμενα παράθυρα. Ο χρήστης καθορίζει τόσο το μέγεθος του παραθύρου, δηλαδή από πόσα κελιά αποτελείται όσο και το ποσοστό των παραθύρων στα οποία θα πρέπει να εμφανίζεται μια μετάβαση για να θεωρηθεί συχνή. Το πλάτος του παραθύρου συμβολίζεται με width(w). Δεδομένης μιας ακολουθίας s και ενός ακεραίου win, συμβολίζουμε W(s, win) το σύνολο όλων των παραθύρων w στην s τέτοια ώστε width(w) = win. Το Σχήμα 5.2 απεικονίζει δύο παράθυρα πλάτους 4.

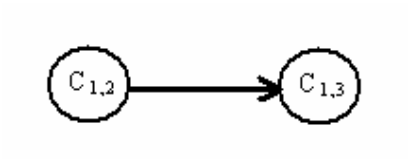


Σχήμα 5.2: Παράθυρα πλάτους 4

Το πρώτο παράθυρο w1 είναι η ακολουθία < $C_{1,2}$, $C_{1,3}$, $C_{4,5}$, $C_{3,5}$> και το δεύτερο w2 είναι η < $C_{1,3}$, $C_{4,5}$, $C_{3,5}$, $C_{2,5}$>. Επίσης υπάρχουν W(s, 4) = 9 παράθυρα πλάτους 4 και παρατηρούμε ότι το κελί $C_{1,2}$ εμφανίζεται σε 6 από τα 9 παράθυρα.

5.1.2 Μεταβάσεις (subtrajectories)

Μια μετάβαση είναι μια συλλογή από κελιά διατεταγμένα στο χρόνο και μπορεί να περιγραφεί από έναν κατευθυνόμενο γράφο. Για παράδειγμα η ακολουθία $C_{1,2}$, $C_{1,3}$ είναι η μετάβαση από το κελί $C_{1,2}$ στο $C_{1,3}$ και μπορεί να περιγραφεί από το γράφο του Σχήματος 5.3.

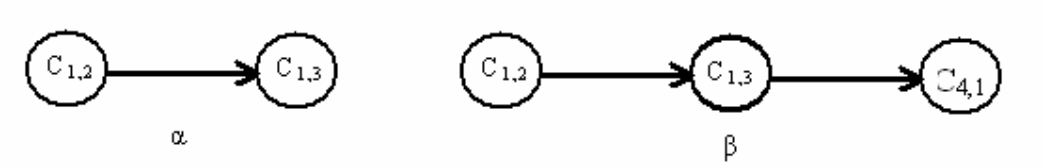


Σχήμα 5.3: Μετάβαση C1,2, C1,3

Τυπικά, μια *μετάβαση* α είναι μια τριπλέτα (V, ≤, g), όπου V είναι ένα σύνολο κόμβων του γράφου, ≤ μια διάταξη των κόμβων, και g: V→E απεικόνιση κάθε κόμβου σε ένα κελί. Τα κελιά πρέπει να έχουν τη διάταξη που ορίζεται από το ≤. Το μέγεθος της α συμβολίζεται με |α| και ισούται με |V|, τον αριθμό δηλαδή των κελιών από τα οποία αποτελείται. Για παράδειγμα, στη μετάβαση που φαίνεται στο Σχήμα 5.3, το σύνολο V περιέχει δύο κόμβους x, y. Η απεικόνιση g αντιστοιχίζει αυτούς

τους κόμβους στα κελιά $C_{1,2}$, $C_{1,3}$ δηλαδή g(x)= $C_{1,2}$ και g(y)= $C_{1,3.}$ Με βάση τη διάταξη ≤, x≤y , το κελί $C_{1,2}$ θα πρέπει να επισκέπτεται από το χρήστη πριν το $C_{1,3.}$

Στη συνέχεια ορίζουμε πότε μια μετάβαση είναι *υποσύνολο* μιας άλλης. Αυτή η σχέση θα χρησιμοποιηθεί στον αλγόριθμο για την ανακάλυψη των συχνών μεταβάσεων. Μια μετάβαση β=(V΄, ≤΄, g΄) είναι υποσύνολο μιας μετάβασης α=(V, ≤, g), και συμβολίζεται με β≤α, όταν υπάρχει απεικόνιση g΄(v)=g(f(v)) για κάθε v ϵ V΄ και για κάθε v,w ϵ V΄ με v≤w΄ επίσης f(v)=f(w). Μια μετάβαση α είναι υπερσύνολο της β αν και μόνο αν β≤α. Γράφουμε β<α αν β≤α και αν δεν ισχύει α≤β. Στο Σχήμα 5.4 βλέπουμε ότι α≤β, αφού το α είναι «υπο-γράφος» του β. Με βάση τον ορισμό και οι δύο κόμβοι του α έχουν αντίστοιχους κόμβους στο β, που έχουν και την ίδια διάταξη.



Σχήμα 5.4: Γράφοι μεταβάσεων

Αν εφαρμόσουμε τα παραπάνω σε μια ακολουθία, τότε λέμε ότι μια μετάβαση *συμβαίνει* στην ακολουθία, όταν οι κόμβοι της μετάβασης έχουν αντίστοιχα κελιά στην ακολουθία με ίδια ετικέτα-όνομα και ίδια διάταξη. Μια μετάβαση α=(V, ≤, g) συμβαίνει σε μια ακολουθία κελιών s = <A_1,A_2,……,A_n>, αν υπάρχει απεικόνιση h: V→{1,……,n} από κόμβους σε κελιά τέτοια ώστε g(x)=$A_{h(x)}$ για όλα τα x ϵ V, και για όλα τα x, y ϵ V με x≠y έχουμε $t_{h(x)} < t_{h(y)}$. Στο Σχήμα 5.2, για παράδειγμα, το παράθυρο w1 περιέχει τα κελιά $C_{1,2}$, $C_{1,3}$, $C_{4,5}$, $C_{3,5}$. Η μετάβαση α συμβαίνει στο w1, όμως η β όχι.

Ορίζουμε τη *συχνότητα* μιας μετάβασης ως το ποσοστό των παραθύρων στα οποία συμβαίνει η μετάβαση αυτή. Δοθείσης μιας ακολουθίας κελιών s και ενός πλάτους παραθύρου win, η συχνότητα μιας μετάβασης α στην s είναι

$$\mathrm{fr}(\alpha, s, \mathrm{win}) = \frac{|\{w \in W(s,\mathrm{win}) / \alpha \text{ occurs in } w\}|}{|W(s,\mathrm{win})|}.$$

Δοθέντος ενός κατωφλίου συχνότητας min_fr,η α είναι συχνή αν fr(α, s, win)≥min_fr. Το ζητούμενο είναι αν μας δοθεί ένα σύνολο από μεταβάσεις να ανακαλύψουμε ποιες από αυτές είναι συχνές. Το σύνολο των συχνών μεταβάσεων που σέβεται τα s, win, min_fr συμβολίζεται με F(s, win, min_fr).

5.2 Αλγόριθμος

Δίνεται μια ακολουθία κελιών s, ένα σύνολο Ε από μεταβάσεις, ένα πλάτος παραθύρου win και ένα κατώφλι συχνότητας min_fr. Σκοπός μας είναι να βρούμε το σύνολο F(s, win, min_fr) των συχνών μεταβάσεων (frequent subtrajectories).

5.2.1 Κύριος αλγόριθμος

Ο αλγόριθμος που φαίνεται στο Σχήμα 5.5 υπολογίζει το σύνολο F(s, win, min_fr). Ο αλγόριθμος εκτελεί μια αναζήτηση των μεταβάσεων κατά επίπεδο. Η αναζήτηση ξεκινά με τις πιο γενικές μεταβάσεις, δηλαδή αυτές που αποτελούνται από ένα κελί. Σε κάθε επίπεδο ο αλγόριθμος υπολογίζει πρώτα μια συλλογή από υποψήφιες μεταβάσεις (candidates) και μετά ελέγχει τις συχνότητές τους από τη ΒΔ. Το κρίσιμο σημείο στην παραγωγή των candidates δίνεται από την παρακάτω παρατήρηση:

Αν μια μετάβαση α είναι συχνή σε μια ακολουθία κελιών s, τότε όλες οι μεταβάσεις που είναι υποσύνολα της α (β≤α) είναι συχνές.

Αλγόριθμος 1
Είσοδος: Ένα σύνολο Ε από κελιά, μια ακολουθία κελιών, ένα σύνολο ξ από μεταβάσεις,
πλάτος παραθύρου win και κατώφλι συχνότητας min_fr.
Έξοδος: Το σύνολο F(s, win, min_fr) των συχνών μεταβάσεων.
Μέθοδος:

1. Υπολόγισε C_1:={α ϵ ξ/ |α|=1};
2. l:=1;
3. **while** $C_l \neq 0$ **do**
4. /*Database pass*/
5. Υπολόγισε το F_l:={ C_l / fr(α, s, win) ≥ min_fr};
6. l:=l+1;
7. /* Candidate generation*/
8. Υπολόγισε C_l := { α ϵ ξ/ |α|=l και για όλα τα β ϵ ξ τέτοια ώστε β<α
9. και |β|<l να έχουμε β ϵ $F_{|\beta|}$};
10. **for** all l **do** output F_l;

Σχήμα 5.5: Κύριος αλγόριθμος

Στις παραγράφους που ακολουθούν περιγράφονται λεπτομερώς οι φάσεις της παραγωγής των υποψήφιων μεταβάσεων και του περάσματος της ΒΔ.

5.2.2 Παραγωγή υποψήφιων μεταβάσεων (candidates generation)

Σε αυτή την ενότητα θα περιγράψουμε τον αλγόριθμο παραγωγής υποψήφιων μεταβάσεων, που φαίνεται στο Σχήμα 5.6. Στον αλγόριθμο κάθε μετάβαση α=(V,≤,g) παριστάνεται σαν ένας πίνακας από κελιά διατεταγμένα στο χρόνο. Για παράδειγμα η μετάβαση του Σχήματος 5.4 αναπαρίσταται από τον πίνακα β[0]=$C_{1,2}$, β[1]=$C_{1,3}$, β[2]=$C_{1,4}$. Συλλογές από μεταβάσεις αναπαρίστανται από ένα διατεταγμένο πίνακα, δηλαδή η i-στη μετάβαση μιας συλλογής F είναι η F[i]. Αφού οι μεταβάσεις και οι συλλογές από μεταβάσεις είναι διατεταγμένες, όλες οι μεταβάσεις που έχουν ίδια τα πρώτα κελιά είναι συνεχείς στις συλλογές μεταβάσεων. Συγκεκριμένα, αν έχουμε δύο

συλλογές από μεταβάσεις $F_l[i]$ και $F_l[j]$ μεγέθους l και αυτές έχουν κοινά τα πρώτα l-1 κελιά, τότε και όλες οι υπόλοιπες συλλογές $F_l[k]$, $i \leq k \leq j$ έχουν επίσης ίδια τα πρώτα l-1 κελιά. Η μέγιστη ακολουθία από συνεχείς μεταβάσεις μεγέθους l, που έχουν κοινά τα πρώτα l-1 κελιά ονομάζεται block. Πιθανές υποψήφιες μεταβάσεις μπορεί να δημιουργηθούν αν κάνουμε όλους τους συνδυασμούς δύο μεταβάσεων στο ίδιο block.

Αλγόριθμος 2
Είσοδος: Διατεταγμένος πίνακας Fl από συχνές μεταβάσεις μεγέθους l.
Έξοδος: Διατεταγμένος πίνακας από υποψήφιες μεταβάσεις μεγέθους l+1.
Μέθοδος:

```
1.  C_{l+1}:=Ø;
2.  k:=0;
3.  if l=1 then for h:=1 to |F_1| do F_1.block_start[h]:=1;
4.  for i:=1 to |F_1| do
5.      current_block_start:=k+1;
6.      for (j:=F_1.block_start[i]; F_1.block_start[j];==F_1.block_start[i];j:=j+1) do
7.          /*τα F_l[i] και F_l[j] που έχουν κοινά τα πρώτα l-1 κελιά, και
8.           σχηματίζουν με τον συνδυασμό τους μια υποψήφια μετάβαση*/
9.           for x:=1 to l do α[x]:=F_l[i][x];
10.          α[l+1]:=F_l[j][l];
11.  /*Δημιουργία υποσυνόλων μεταβάσεων β που δεν περιέχονται στο α[y]*/
12.          for y:=1 to l-1 do
13.              for x:=1 to y-1 do β[x]:=α[x];
14.              for x:=y to l do β[x]:=α[x+1];
15.              If β is not in Fl then continue with next j at line 6;
16.      /*όλα τα υποσύνολα μεταβάσεων στο F_l, αποθηκεύουν την α σαν
17.      υποψήφια μετάβαση*/
18.          k:=k+1;
19.          C_{l+1}[k]:=α
20.          C_{l+1}.block_start[k]:=current_block_start;
21. output C_{l+1}
```

Σχήμα 5.6: αλγόριθμος παραγωγής υποψήφιων μεταβάσεων

5.2.3 Αναγνώριση μεταβάσεων σε ακολουθίες

Σε αυτή την ενότητα περιγράφεται ο αλγόριθμος που πραγματοποιεί το πέρασμα της ΒΔ, για να αναγνωρίσει μεταβάσεις στις ακολουθίες. Για δύο παράθυρα $w=(w,t_s,t_s+win)$ και $w'=(w',t_s+1,t_s+win+1)$, όπου t_s η στιγμή που ξεκινά το παράθυρο, οι ακολουθίες w και w΄ είναι παρόμοιες. Επωφελούμενοι από αυτή την ομοιότητα, αφού αναγνωρίσουμε τις μεταβάσεις στο w, κάνουμε αυξητικές ενημερώσεις για να

πετύχουμε την ολίσθηση του παραθύρου και να πάρουμε το w΄. Όταν υπολογίζουμε τη συχνότητα των μεταβάσεων, λαμβάνονται υπ' όψιν φυσικά μόνο τα παράθυρα που υπάρχουν στην ακολουθία εισόδου. Οι υποψήφιες μεταβάσεις αναγνωρίζονται σε μια ακολουθία κελιών χρησιμοποιώντας αυτόματα κατάστασης που δέχονται τις υποψήφιες μεταβάσεις και αγνοούν όλες τις άλλες εισόδους. Η ιδέα είναι ότι για κάθε μετάβαση υπάρχει ένα αυτόματο. Αρχικοποιούμε ένα καινούριο στιγμιότυπο του αυτόματου για μια μετάβαση α κάθε φορά που το πρώτο κελί του α εισέρχεται στο παράθυρο. Το αυτόματο απομακρύνεται όταν το ίδιο γεγονός βγαίνει απ΄ το παράθυρο. Όταν ένα αυτόματο για την α φτάνει στην αποδεκτή κατάσταση, δείχνοντας ότι η α συμπεριλαμβάνεται εξ' ολοκλήρου στο παράθυρο και δεν υπάρχουν ήδη στην αποδεκτή κατάσταση άλλα αυτόματα για την α, αποθηκεύουμε τη στιγμή έναρξης του παραθύρου που περιλαμβάνει την α. όταν απομακρυνθούν όλα τα αυτόματα για την α, τότε αυξάνουμε τη συχνότητα της α κατά τον αριθμό των παραθύρων στα οποία η α εμφανίστηκε ολόκληρη.

Αλγόριθμος 3
Είσοδος: Διατεταγμένος πίνακας Fl από συχνές μεταβάσεις μεγέθους l.
Έξοδος: Διατεταγμένος πίνακας από υποψήφιες μεταβάσεις μεγέθους l+1.
Μέθοδος:

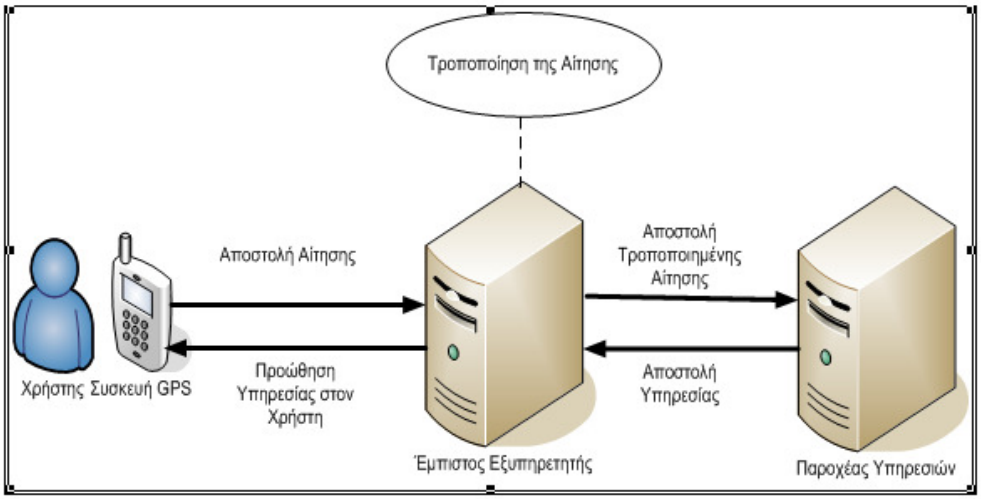
```
1.  /*Αρχικοποίηση*/
2.  for each α in C do
3.      for i:=1 to α do
4.          α.initialized[i]:=0;
5.          waits(α[i]):= Ø;
6.  for each α ϵ C do
7.      waits(α[1]):= waits(α[1])U{(α,1)};
8.      α.frequent_count:=0;
9.  for t:=Ts-win to Ts-1 do beginsat(t):= Ø;
10. /*Αναγνώριση*/
11. for start:=Ts-win+1 to Te do
12.      /*Φέρνουμε νέα κελιά στο παράθυρο*/
13.     beginsat(start+win-1):= Ø;
14.     transitions:= Ø;
15.     for all events (A,t) in s such that t=start+win-1 do
16.         for all (α,j) ϵ waits(A) do
17.             if j=|α| και α.initialized[j]:=0 then α.inwindow:=start;
18.             if j=1 then
19.                 transitions:=transitions U {(α,1,start+win-1)};
20.             else
21.                 transitions:=transitions U {(α,j, α.initialized[j-1])};
22.                 beginsat(α.initialized[j-1]):=
23.                         beginsat(α.initialized[j-1])\{(α,j-1)};
24.                 α.initialized[j-1]:=0;
25.                 waits(A):=waits(A)\{(α,j)};
26.         for all (α,i,t) ϵ transitions do
27.         α.initialized[j]:=t;
28.         beginsat(t):=beginsat(t) U {(α,j)};
29.         if j<|α| then waits(α[j+1]):= waits(α[j+1])  U {(α,j+1)};
30.     /*Εξοδος παλιών γεγονότων από το παράθυρο */
31.     for all (α,l) ϵ beginsat(start-1) do
32.         if l=|α| then α.frequent_count:= α.frequent_count-α.window+start;
33.         else waits(α[l+1]):= waits(α[l+1])\ {(α,l+1)};
34.         α.initialized[l]:=0;
35. /*Output*/
36.  for all episodes α in C do
37.      if α.frequent_count/Te-Ts+win-1≥min_fr then output α;
```

Σχήμα 5.7: Αλγόριθμος αναγνώρισης μεταβάσεων σε ακολουθίες

ΚΕΦΑΛΑΙΟ 6

6. Ιδιωτικότητα χώρο – χρονικών αντικειμένων

Στα προηγούμενα κεφάλαια είδαμε πώς μπορούμε να δημιουργήσουμε ένα οδικό δίκτυο και να μοντελοποιήσουμε τις κινήσεις των κινουμένων αντικειμένων στο χώρο και στο χρόνο επάνω στο δίκτυο αυτό. Τα κινούμενα αντικείμενα είναι κάτοικοι της πόλης του Oldenburg που δημιουργήσαμε και ενώ μετακινούνται για μια χρονική περίοδο T καταγράφονται οι τροχιές τους. Ανά τακτά χρονικά διαστήματα κάθε χρήστης μπορεί να στέλνει στον πάροχο υπηρεσιών (Service Provider) μέσω ειδικής συσκευής (π.χ. GPS) μια αίτηση για την παροχή κάποιας LBS υπηρεσίας. Το ζήτημα που προκύπτει είναι κατά την επικοινωνία αυτή του χρήστη με τον SP να διατηρείται η ιδιωτικότητα του χρήστη, να μην ανταλλάσσονται δηλαδή στοιχεία που θα μπορούσαν να αποκαλύψουν την ταυτότητα του. Όπως αναφέραμε και στο δεύτερο κεφάλαιο, το μοντέλο επικοινωνίας που χρησιμοποιούμε είναι αυτό που φαίνεται στο Σχήμα 2.1 και στο Σχήμα 6.1. Κάθε χρήστης στέλνει την αίτησή του στον παροχέα και αυτός τη στέλνει σε έναν έμπιστο εξυπηρετητή (Trusted Server). Εκεί η αίτηση μετασχηματίζεται κατάλληλα με ειδικές διαδικασίες ώστε να είναι ασφαλής και έπειτα στέλνεται στον SP όπου επεξεργάζεται για να παραχθούν οι ζητούμενες υπηρεσίες. Σε αυτό το κεφάλαιο θα αναλύσουμε τις βασικές έννοιες του προβλήματος της ιδιωτικότητας των κινουμένων αντικειμένων στις οποίες στηρίζεται η λειτουργία του παραπάνω μοντέλου.



Σχήμα 6.1: Μοντέλο προστασίας της ιδιωτικότητας

6.1 Προσδιοριστές προστασίας

Σε μια Βάση Δεδομένων μπορεί να είναι αποθηκευμένα στοιχεία που είναι μοναδικά για κάθε χρήστη, όπως το ΑΦΜ του και ο αριθμός ταυτότητάς του και τα οποία μπορεί αν συνδυαστούν με άλλες πληροφορίες να αποκαλύψουν την ταυτότητα του χρήστη. Τα στοιχεία αυτά ονομάζονται προσδιοριστές προστασίας. Στην παρούσα εργασία θεωρούμε ως προσδιοριστές προστασίας τα στοιχεία που βασίζονται σε χώρο – χρονικά δεδομένα και συγκεκριμένα στις συχνές διαδρομές που ακολουθεί ο χρήστης. Κάθε χρήστης χρησιμοποιεί κινητό ή κάποια άλλη συσκευή με ενσωματωμένο GPS, την οποία μπορεί να έχει συνεχώς ενεργοποιημένη. Εάν ο παροχέας έχει παρακολουθήσει με κάποιο τρόπο τις κινήσεις κάποιου χρήστη και έχει καταγράψει τις συχνές διαδρομές που έχει ακολουθήσει, τότε η ταυτότητά του μπορεί να αποκαλυφθεί. Έτσι, θα πρέπει να προστατευθεί η ιδιωτικότητα και η ασφάλεια κάθε χρήστη.

Παραδείγματα προσδιοριστών προστασίας

Θεωρούμε ένα χρήστη που κάνει τη διαδρομή Σπίτι – Γραφείο πέντε φορές την εβδομάδα. Κάποιες μέρες εργάζεται το πρωί, οπότε ξεκινά από το σπίτι του στις 8:00 π.μ. και φτάνει στο γραφείο στις 9:00 π.μ. και κάποιες μέρες εργάζεται το απόγευμα, οπότε ξεκινά από το σπίτι του στις 3:00 μ.μ. και φτάνει στο γραφείο στις 4:00 μ.μ. Εάν ο μόνος περιορισμός που θέσουμε, για να θεωρήσουμε τη διαδρομή συχνή, είναι η συχνότητα εκτέλεσης της, δηλαδή αν θεωρούμε μια διαδρομή συχνή όταν εκτελείται πάνω από τέσσερις φορές, ανεξαρτήτως χρονικών περιορισμών, τότε η παραπάνω διαδρομή θεωρείται συχνή.

Στο δεύτερο παράδειγμα θεωρούμε έναν οδηγό ταξί ο οποίος επαναλαμβάνει έξι φορές σε μια μέρα τη διαδρομή Θησείο – Σύνταγμα. Αυτή η διαδρομή θεωρείται συχνή, γιατί δεδομένου ότι είναι μικρή η πιθανότητα να κάνουν την ίδια διαδρομή τις ίδιες στιγμές άλλοι συνάδελφοι του, είναι πιθανό να αποκαλυφθεί η ταυτότητά του.

Τέλος θεωρούμε ένα πλοίο που κάνει το ταξίδι Ελλάδα – Τυνησία τέσσερις φορές μέσα σε ένα χρόνο. Και αυτή η διαδρομή θεωρείται συχνή, δεδομένου ότι το πλοίο δεν είχε το χρόνο να κάνει και άλλες διαδρομές και ότι δεν υπάρχουν πολλά πλοία που εκτελούν αυτό το δρομολόγιο.

Στα παραδείγματα που αναφέραμε παρατηρούμε ότι δεν είναι απαραίτητο να υπάρχουν χρονικοί περιορισμοί για να θεωρηθεί προσδιοριστής προστασίας μια διαδρομή που ακολουθεί ένας χρήστης. Οι διαδρομές που εκτελούνται μόνο μια φορά ή σπάνια δε μας απασχολούν, γιατί είναι πολύ δύσκολο τέτοιου είδους διαδρομές να αποκαλύψουν προσωπικά δεδομένα του χρήστη - αιτούντα. Αν ένας χρήστης επισκέπτεται την καθολική εκκλησία στην πόλη του μια φορά το μήνα και αυτό αποκαλυφθεί μέσω αιτήσεων που στέλνει, τότε μπορούμε να συμπεράνουμε ποιο

είναι το θρήσκευμά του οπότε κινδυνεύει η ιδιωτικότητά του. Αν αυτή η διαδρομή γίνεται μια φορά το χρόνο, τότε δεν μπορεί εύκολα να συνδεθεί με προηγούμενες διαδρομές του χρήστη, οπότε δεν κινδυνεύει η ιδιωτικότητά του. Σύμφωνα με τα παραπάνω ένας χρήστης μπορεί να έχει περισσότερους από έναν προσδιοριστές προστασίας. Ο αριθμός τους εξαρτάται από τις διαδρομές που εκτελεί ο χρήστης. Οι προσδιοριστές προστασίας όπως και όλο το ιστορικό των κινήσεων του χρήστη, αποθηκεύονται στον έμπιστο εξυπηρετητή (TS). Παρακάτω παραθέτουμε έναν πιο ακριβή ορισμό για το τι θεωρούμε προσδιοριστή προστασίας, τον οποίο διαβάσαμε στο [3].

Ορισμός 6.1: Ένας προσδιοριστής προστασίας είναι ένα χωρικό πρότυπο μετακίνησης, που αποτελείται από μια ακολουθία χωρικών στοιχείων και από έναν τύπο επανάληψης. Κάθε χωρικό στοιχείο είναι της μορφής <Περιοχή>. Ο τύπος επανάληψης είναι εκείνο το στοιχείο που καθορίζει αν μια ακολουθία από χωρικά στοιχεία είναι προσδιοριστής προστασίας.

Τα χωρικά στοιχεία όπως αναφέρουμε στον ορισμό είναι της μορφής <Περιοχή >. Η περιοχή είναι ένα «κομμάτι» της συνολικής περιοχής στην οποία έχουμε ορίσει ότι μπορεί να κινείται ο χρήστης και ονομάζεται, όπως αναφέραμε στο τρίτο κεφάλαιο, κελί. Η περιοχή αυτή(κελί) είναι ένα χωρικό δεδομένο και αναπαρίσταται από μια ετικέτα $C_{i,j}$, που δείχνει το κελί στο οποίο ανήκουν οι συντεταγμένες (x, y) του σημείου όπου βρίσκεται ο χρήστης. Ο τύπος επανάληψης είναι εκείνο το στοιχείο που καθορίζει αν μια ακολουθία από χωρικά στοιχεία είναι προσδιοριστής προστασίας. Πιο συγκεκριμένα καθορίζει πόσες φορές πρέπει να επαναληφθεί μια διαδρομή για να θεωρηθεί προσδιοριστής προστασίας. Αν ένας χρήστης εκτελεί μια διαδρομή όσες φορές ορίζει ο τύπος επανάληψης, τότε αυτή θεωρείται προσδιοριστής προστασίας. Ο

τύπος αυτός αποτελεί ένα είδος χρονικού περιορισμού. Σύμφωνα με το [3], η σύνταξη αυτού του τύπου είναι η εξής:

$$\mathbf{r_1.G_1 * r_2.G_2 *\ldots.* r_{k-1}.G_{k-1} * r_k.G_k} \quad , \quad \text{με} \quad k=0, 1, \ldots. n$$

Το G_k συμβολίζει το χρονικό διάστημα μέσα στο οποίο θα πρέπει να παρατηρηθεί η επανάληψη της διαδρομής και μπορεί να είναι μήνες, βδομάδες, μέρες. Το G_k είναι μια διαβάθμιση του χρόνου. Το r_k είναι ένας ακέραιος που ορίζει πόσες φορές εμφανίζεται μια χωρική ακολουθία <Περιοχή> σε κάθε διάστημα G_k. Σύμφωνα με τον τύπο αυτό, μια ακολουθία πρέπει να εμφανίζεται τουλάχιστον $\mathbf{r_1}$ φορές εντός του $\mathbf{G_1}$, που είναι το μικρότερο διάστημα, $\mathbf{r_2}$ φορές εντός του $\mathbf{G_2}$, που είναι το αμέσως μεγαλύτερο διάστημα και τελικά $\mathbf{r_n}$ φορές εντός του $\mathbf{G_n}$, που είναι το μεγαλύτερο διάστημα. Για τους σκοπούς της εργασίας μας δε θα χρησιμοποιήσουμε επακριβώς τον τύπο επανάληψης. Θα ορίσουμε μόνο έναν ελάχιστο αριθμό φορών στις οποίες θα πρέπει να εκτελείται μια διαδρομή για να θεωρηθεί συχνή.

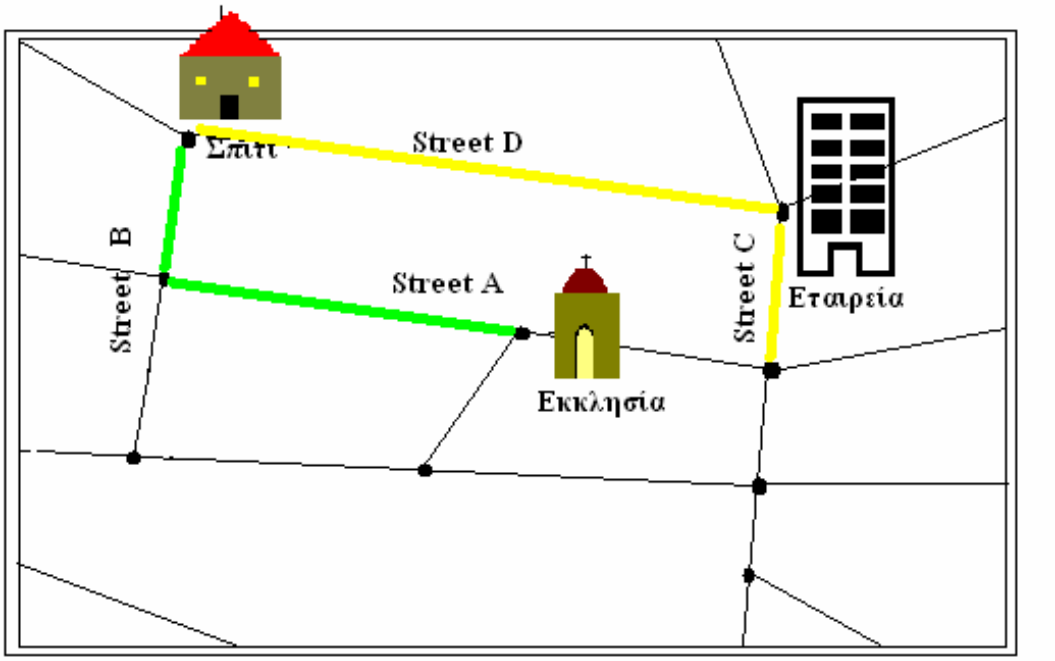
Τελικά, με βάση τα παραπάνω ένας προσδιοριστής προστασίας έχει τη μορφή

$$\mathbf{<E_1, E_2,\ldots.E_n>, r_1.G_1 * r_2.G_2 *\ldots.* r_{k-1}.G_{k-1} * r_k.G_k},$$

όπου E_k τα χωρικά στοιχεία, που όπως είπαμε παραπάνω έχουν τη μορφή <Περιοχή> και η περιοχή αποτελείται από τη διαδρομή που εκτελεί ο χρήστης.

Για την κατανόηση της έννοιας των προσδιοριστών προστασίας, δίνουμε ένα παράδειγμα στο Σχήμα 6.2. Στο σχήμα αυτό απεικονίζεται ένα μικρό τμήμα μιας πόλης στο οποίο βρίσκονται το σπίτι ενός χρήστη, η εταιρεία στην οποία εργάζεται και η καθολική εκκλησία. Η πράσινη και η κίτρινη διαδρομή είναι διαδρομές που εκτελεί συχνά ο χρήστης. Η κίτρινη διαδρομή είναι η διαδρομή Σπίτι – Εταιρεία η

οποία γίνεται πέντε φορές την εβδομάδα και η πράσινη διαδρομή Σπίτι – Εκκλησία που γίνεται τέσσερις φορές μέσα σε ένα μήνα. Έτσι ένας προσδιοριστής προστασίας αυτού του παραδείγματος είναι ο εξής: <<Street D, Street C>, <Street A, Street B>>. Αν ο χρήστης υποβάλλει κάποια αίτηση από αυτούς τους δρόμους, τότε θα πρέπει η αίτηση αυτή να προστατευθεί.



Σχήμα 6.2: Παράδειγμα προσδιοριστή προστασίας

Ορισμός 6.2: Αν ένας χρήστης κάνει μια αίτηση r_k από τη (x_i, y_j), η οποία βρίσκεται στο κελί $C_{i,j}$, τότε λέμε ότι η αίτηση r_k ταιριάζει με ένα χωρικό στοιχείο E_k του προσδιοριστή προστασίας, αν το κελί $C_{i,j}$ περιέχεται στην τροχιά που έχουμε ορίσει στο E_k .

Με τον ίδιο ακριβώς τρόπο μπορούμε να ελέγξουμε αν ένα σύνολο από αιτήσεις ταιριάζουν με κάποιον από τους προσδιοριστές προστασίας.

Ορισμός 6.3: Ένα σύνολο αιτήσεων R λέμε ότι ταιριάζει με ένα στοιχείο E_k του προσδιοριστή προστασίας, αν ισχύουν τα παρακάτω:

1) Κάθε αίτηση r_k του συνόλου αιτήσεων R ταιριάζει με ένα στοιχείο E_k .

2) Οι χρονικές που έγιναν όλες οι αιτήσεις του συνόλου R (δηλαδή κάθε αίτηση r_k έγινε τη χρονική στιγμή t_k), ικανοποιούν όλες τον τύπο επανάληψης.

6.2 Προσωπικό Ιστορικό Τοποθεσιών

Κατά τη διάρκεια της κίνησης του κάθε χρήστης στέλνει στον SP (παροχέα υπηρεσιών) ενημερώσεις για τη θέση στην οποία βρίσκεται και τη χρονική στιγμή την οποία βρίσκεται σε κάθε θέση. Οι ενημερώσεις αυτές ονομάζονται ενημερώσεις θέσης (location updates). Η ακολουθία των location updates συνιστά το ιστορικό τοποθεσιών κάθε χρήστη (PHL), το οποίο αποθηκεύεται στον έμπιστο εξυπηρετητή (TS), και δηλώνει ποια σημεία επισκέφθηκε ο χρήστης και ποια στιγμή επισκέφθηκε το κάθε σημείο. Αφού οι χρήστες κινούνται διαρκώς, τα PHL τους αλλάζουν. Οι αλλαγές αυτές γίνονται από τον TS στον οποίο και είναι αποθηκευμένα τα PHL και ο οποίος γνωρίζει όλες τις κινήσεις των χρηστών. Στο [3] δίνεται ο παρακάτω ορισμός για τα PHL.

Ορισμός 6.4: Το PHL ενός χρήστη είναι μια ακολουθία από χώρο –χρονικά στοιχεία, τα οποία έχουν την εξής μορφή: (x, y, t). Τα x, y παριστάνουν τις ακριβείς συντεταγμένες του σημείου από το οποίο πέρασε ο χρήστης, ενώ το t είναι η ακριβής χρονική στιγμή που ο χρήστης βρέθηκε στο σημείο με συντεταγμένες (x, y).

Ορισμός 6.5: Έστω ότι έχουμε ένα σύνολο αιτήσεων $\mathbf{R=(r_1,r_2,….,r_n)}$ και ένα PHL ενός χρήστη. Το PHL του χρήστη λέμε ότι είναι χώρο –χρονικά συνεπές με το σύνολο των αιτήσεων **R** αν ισχύει **μία** από τις παρακάτω περιπτώσεις:

1) Για κάθε αίτηση $\mathbf{r_k\ (x_k, y_k, t_k)}$ του συνόλου **R**, υπάρχει ένα στοιχείο $\mathbf{(x_l, y_l, t_l)}$ στο **PHL** του χρήστη τέτοιο ώστε: $\mathbf{x_k=x_l\ \&\ y_k=y_l\ \&\ t_k=t_l.}$

2) Για κάθε αίτηση $\mathbf{r_k\ (x_k, y_k, t_k)}$ του συνόλου **R**, υπάρχουν δύο στοιχεία $\mathbf{(x_l, y_l, t_l)}$ και $\mathbf{(x_m, y_m, t_m)}$ στο **PHL** του χρήστη τέτοια ώστε:

$$\mathbf{x_l \le x_k \le x_m \quad \& \quad y_l \le y_k \le y_m \quad \& \quad t_l \le t_k \le t_m.}$$

6.3 Διασύνδεση μεταξύ χρηστών – αιτήσεων

Με τον όρο διασύνδεση μεταξύ χρηστών – αιτήσεων, εννοούμε τη συσχέτιση που μπορεί να υπάρξει μεταξύ ενός χρήστη με τις αιτήσεις που θα κάνει στο μέλλον. Αυτή η συσχέτιση είναι επικίνδυνη για την ιδιωτικότητα του χρήστη, αφού μπορεί, αν είναι γνωστό ότι ανήκει σε αυτόν κάποια αίτηση, να αποκαλυφθεί η ταυτότητά του. Για να μη γίνεται η αντιστοίχηση των αιτήσεων στους χρήστες που τις πραγματοποιούν, έχουν προταθεί διάφορες τεχνικές ώστε να προστατευθεί η ιδιωτικότητά τους. Μια από αυτές τις τεχνικές είναι η αλλαγή των αναγνωριστικών των χρηστών σε τακτά χρονικά διαστήματα. Τα μειονεκτήματα αυτής της τεχνικής είναι τα εξής: 1) αν η αλλαγή των αναγνωριστικών γίνεται πολύ συχνά τότε μπορεί να υπάρξει σύνδεση μεταξύ κάποιων αιτήσεων του χρήστη και 2) η τεχνική αυτή έχει μεγάλο κόστος, γιατί κάθε φορά που ο χρήστης αλλάζει αναγνωριστικό θα πρέπει στη

Βάση Δεδομένων με τα PHL και τους προσδιοριστές προστασίας να αντικατασταθεί το παλιό του αναγνωριστικό με το καινούριο.

Στην παρούσα εργασία θα εφαρμόσουμε μια τεχνική που βασίζεται σε πιθανοτικές συναρτήσεις και μοντέλα για τη διασύνδεση μεταξύ χρηστών – μελλοντικών αιτήσεων. Αν η πιθανότητα που επιστρέφει η τεχνική είναι μεγάλη, τότε θα πρέπει να εφαρμοστούν επιπλέον τεχνικές ώστε να προστατευθεί η ιδωτικότητα του χρήστη. Συγκεκριμένα θεωρούμε ότι ο TS διαθέτει ένα σύνολο από συναρτήσεις διασύνδεσης, το **LINK(),** οι οποίες επιστρέφουν την πιθανότητα που υπάρχει να μπορεί να συσχετιστεί ένας χρήστης με τις μελλοντικές του αιτήσεις. Οπότε οι τιμές που επιστρέφει η LINK ανήκουν στο διάστημα [0,1]. Έστω ότι έχουμε δύο αιτήσεις ($\mathbf{r_i}$ και $\mathbf{r_j}$), τότε για τη LINK ισχύουν οι παρακάτω ιδιότητες:

1) $\mathbf{LINK(r_i, r_j) = LINK(r_j, r_i)}$, που σημαίνει ότι η LINK είναι συμμετρική.

2) $\mathbf{LINK(r_i, r_j) = 1 \quad \& \quad LINK(r_j, r_i) = 1}$, που σημαίνει ότι η LINK είναι ανακλαστική.

Αν η LINK επιστρέψει 1, που είναι και η μέγιστη τιμή που μπορεί να επιστρέψει, αυτό σημαίνει ότι οι αιτήσεις $\mathbf{r_i}$ και $\mathbf{r_j}$ έχουν σταλεί από τον ίδιο χρήστη, οπότε κινδυνεύει η ιδιωτικότητά του και θα πρέπει να γίνει η κατάλληλη διαδικασία ώστε να προστατευθεί. Αν επιστρέψει 0, που είναι και η ελάχιστη τιμή που μπορεί να επιστρέψει, αυτό σημαίνει ότι οι αιτήσεις $\mathbf{r_i}$ και $\mathbf{r_j}$ έχουν σταλεί από διαφορετικούς χρήστες, οπότε δεν κινδυνεύει η ιδιωτικότητά τους και δεν είναι απαραίτητο να γίνουν επιπλέον διαδικασίες ώστε να προστατευθεί.

Ορισμός 6.6: Έστω ότι έχουμε ένα σύνολο αιτήσεων $\mathbf{R=(r_1,r_2,\ldots,r_n)}$ που έχουν σταλεί στον έμπιστο εξυπηρετητή. Τότε η τιμή που επιστρέφει η συνάρτηση

LINK(R) είναι η πιθανότητα δύο διαφορετικές αιτήσεις $\mathbf{r_i}$ και $\mathbf{r_j}$ (του συνόλου **R**) να έχουν σταλεί από τον ίδιο χρήστη.

Ορισμός 6.7: Έστω ότι έχουμε ένα σύνολο αιτήσεων $\mathbf{R=(r_1,r_2,\ldots,r_n)}$ που έχουν σταλεί στον έμπιστο εξυπηρετητή και ένα υποσύνολο **R΄** του **R**. Τότε λέμε ότι το σύνολο **R΄** διασυνδέεται με πιθανότητα **Θ**, αν για κάθε ζεύγος αιτήσεων $\mathbf{(r_i , r_j) \epsilon R΄}$, υπάρχει ένα σύνολο αιτήσεων $\mathbf{r_{i1}, r_{i2}, r_{i3},\ldots\ldots, r_{ik} \epsilon R΄}$, όπου $\mathbf{r_{i1} = r_i}$ και $\mathbf{r_{ik} = r_j}$, τέτοια ώστε $\mathbf{LINK(r_{i1}, r_{i1+1}) \geq \Theta}$ για όλα τα i=1,2,3,….k-1.

Με βάση τον παραπάνω ορισμό, οι αιτήσεις ενός συνόλου $\mathbf{R΄ C R}$, έχουν σταλεί από τον ίδιο χρήστη, αν και μόνο αν το **R΄** διασυνδέεται με πιθανότητα $\mathbf{\Theta = 1.}$

6.4 Εισαγωγή των PHL στην Oracle

Στο Κεφάλαιο 3 αναλύσαμε πώς δημιουργείται ένα χωρικό δίκτυο στο ΣΔΒΔ της Oracle και πώς εισάγονται σε αυτό τα κινούμενα χώρο – χρονικά αντικείμενα. Το προσωπικό ιστορικό τοποθεσιών (PHL) κάθε χρήστη είναι ουσιαστικά τα χώρο – χρονικά δεδομένα που είναι ήδη αποθηκευμένα στη Βάση Δεδομένων (Κεφ. 3.4).

ΚΕΦΑΛΑΙΟ 7

7. Αξιολόγηση της τεχνικής

Στο παρόν Κεφάλαιο θα κάνουμε μια αξιολόγηση της τεχνικής που χρησιμοποιήσαμε για να βρούμε τις συχνές διαδρομές που ακολουθεί κάθε χρήστης. Για το σκοπό αυτό κάναμε πειράματα τρέχοντας τον αλγόριθμο για διάφορες τιμές. Τα πειράματα έγιναν σε ηλεκτρονική υπολογιστή με επεξεργαστή Intel Core 2 Quad Core 2.4 GHz, 4GB μνήμη RAM σε λειτουργικό Windows XP Professional. Το δίκτυο στο οποίο κινούνται οι χρήστες είναι το οδικό δίκτυο της πόλης του Oldenburg, που δημιουργήσαμε με το Generator του Brinkhoff και περιγράψαμε αναλυτικά στο Κεφάλαιο 3.

7.1 Πειραματικά δεδομένα

Για τη διεξαγωγή των πειραμάτων χρησιμοποιήσαμε ένα σύνολο δεδομένων που προκύπτει από την εκτέλεση του Generator του Brinkhoff για το δίκτυο Oldenburg.

- Σύνολο Δεδομένων 1: 100 χρήστες, T=[0,20], 10T

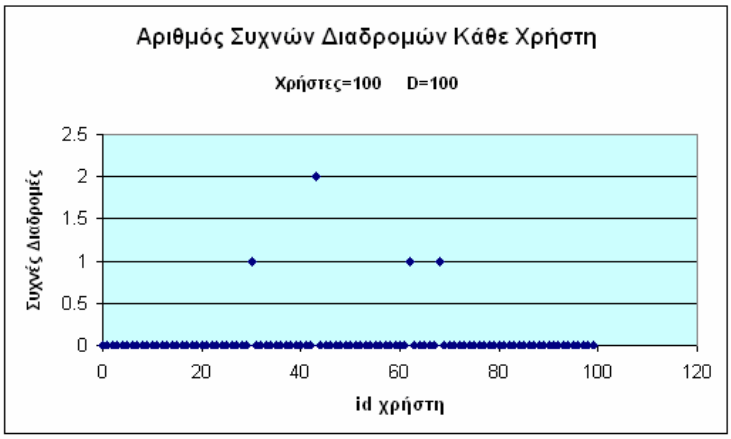
Καθένα από τα παραπάνω σύνολα δεδομένων θα μπορούσαμε να πούμε ότι εκφράζει μια διαφορετική ώρα της ημέρας για δέκα μέρες. Το πρώτο σύνολο για παράδειγμα

αναπαριστά την κίνηση 100 χρηστών, το πολύ για είκοσι ώρες της ημέρας για δέκα μέρες.

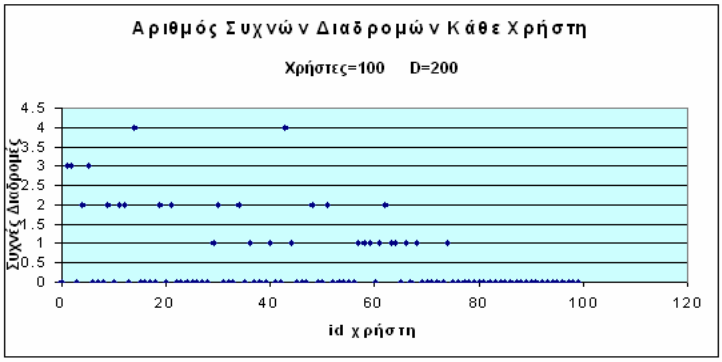
7.2 Πειραματικά αποτελέσματα

Στην ενότητα αυτή θα παρουσιάσουμε τα αποτελέσματα που προέκυψαν από τα πειράματα που κάναμε τρέχοντας τον αλγόριθμο εύρεσης συχνών διαδρομών. Εκτελέσαμε τον αλγόριθμο για τα δύο σύνολα δεδομένων και παριστάνουμε σε γραφήματα τα αποτελέσματα που πήραμε. Τα γραφήματα αποτελούνται από σημεία που δείχνουν πόσες συχνές διαδρομές προκύπτουν για κάθε χρήστη με τον αλγόριθμο. Όπως είπαμε στο Κεφάλαιο 4, για να βρούμε τη διαδρομή που κάνει κάθε χρήστης, χωρίζουμε τους άξονες σε ίσα διαστήματα. Τα παρακάτω γραφήματα παρουσιάζουν τα αποτελέσματα αν χωρίσουμε κάθε άξονα σε διαστήματα μήκους 100, 200 και 500, που τα συμβολίζουμε με D.

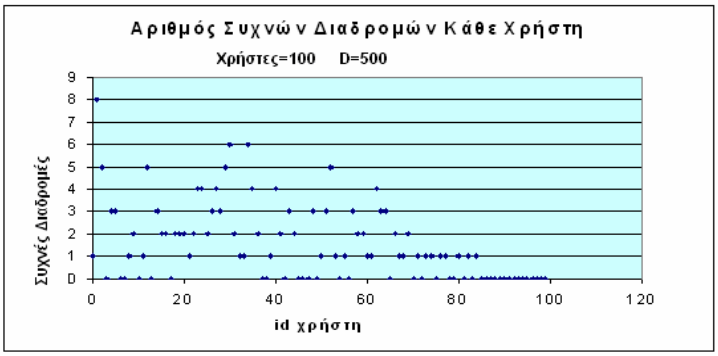
ΣΥΝΟΛΟ ΔΕΔΟΜΕΝΩΝ



Σχήμα 7.1: Σύνολο Δεδομένων 1, D=100



Σχήμα 7.2: Σύνολο Δεδομένων 1, D=200



Σχήμα 7.3: Σύνολο Δεδομένων 1, D=500

7.3 Συμπεράσματα

Από τα παραπάνω γραφήματα μπορούμε εύκολα να συμπεράνουμε πως όσο αυξάνουμε το μήκος των διαστημάτων στα οποία χωρίζουμε τους άξονες x και y, τόσο αυξάνονται και οι συχνές διαδρομές κάθε χρήστη, που αποτελούν τους προσδιοριστές προστασίας. Το βασικό μειονέκτημα είναι ότι το μεγαλύτερο ποσοστό των συχνών διαδρομών, είναι διαδρομές που γίνονται μέσα στο ίδιο κελί και όχι από ένα κελί σε κάποιο άλλο. Έτσι, βελτιώσαμε κατά κάποιο τρόπο το αποτέλεσμα που παράγει ο Generator του Brinkhoff, ο οποίος ενώ δεν μπορεί να παράγει συχνές διαδρομές για τους χρήστες ούτε επαναλαμβάνει σημεία από τα οποία έχει ήδη περάσει ο χρήστης, τροποποιώντας κατάλληλα το αρχείο εξόδου του και δημιουργώντας εμείς τις συχνές διαδρομές.

ΚΕΦΑΛΑΙΟ 8

8. Επίλογος

Στην παρούσα εργασία έχουμε παρουσιάσει τα εξής: Αρχικά περιγράψαμε πώς μπορούμε να δημιουργήσουμε ένα χωρικό (οδικό) δίκτυο, που να αναπαριστά μια πραγματική γεωγραφική περιοχή, με τις δυνατότητες που προσφέρει η Spatial Oracle. Έπειτα με χρήση του Generator του Brinkhoff εισήγαμε τα κινούμενα χώρο – χρονικά αντικείμενα στο δίκτυο και αναπαραστήσαμε τις κινήσεις των χρηστών στο χάρτη. Αφού δημιουργήσαμε τον πίνακα με τις τροχιές των χρηστών, τον τροποποιήσαμε κατάλληλα ώστε να δημιουργούνται συχνές διαδρομές. Στη συνέχεια αναλύσαμε έναν αλγόριθμο με τον οποίο βρίσκουμε τις συχνές διαδρομές και εξηγήσαμε πως μπορούν οι συχνές διαδρομές να θεωρηθούν προσδιοριστές προστασίας και πως σχετίζονται όλα αυτά με το πρόβλημα της ιδωτικότητας στην εξόρυξη τροχιών κινουμένων αντικειμένων. Τις βασικές έννοιες του προβλήματος της ιδιωτικότητας τις περιγράψαμε στο Κεφάλαιο 6.

9. Βιβλιογραφία

[1] Aris Gkoulalas-Divanis and Vassilios S. Verykios. A Free Terrain Model for Trajectory K–Anonymity. In proceedings of 19th International Conference on Database and Expert Systems Applications, pp.49-56. 2008

[2] S. Giannakopoulos. An algorithm for protecting privacy in spatiotemporal data. BS Thesis. Dept. of Computer & Communication Engineering., University of Thessaly, Volos, Greece. 2007.

[3] Dimitris Douras. K-Anonymity in Spatio-Temporal Data Bases BS Thesis. Dept. of Computer & Communication Engineering., University of Thessaly, Volos, Greece. 2008.

[4] P. Samarati and L. Sweeney. Protecting privacy when disclosing information: K-Anonymity and its enforcement through generalization and suppression. Technical Report SRI-CSL-98-04, Computer Science Laboratory, SRI International, 1998.

[5] L. Sweeney. K-Anonymity: A model for protecting privacy. In Proceedings of International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, (Vol. 10, No. 5), 2002.

[6] T. Brinkhoff. A Framework for Generating Network-Based Moving Objects. GeoInformatica, (Vol. 6 No.2) pp.153-180, 2002.

[7] G. Gidofalvi and T. B. Pedersen. Mining long sharable patterns in Trajectories of Moving Objects. In Proc. of STDBM, 2006.

[8] G. Gidofalvi, X. Huang, and T. B. Pedersen. Privacy preserving Data Mining on Moving Objects Trajectories. In Proc. of MDM, 2007.

[9] C. Bettini, X. S. Wang, and S. Jajodia. Protecting Privacy Against Location-Based Personal Identification, in Proceedings of Very large data bases Workshop on Secure Data Management, 2005, pp. 185–199.

[10] C. Bettini, S. Jajodia, X. S. Wang. Time Granularities in Databases, Data Mining and Temporal Reasoning. Springer-Verlag New York, Inc. 2000.

[11]Heiki Mannila, Hannu Toivonen, and A. Inkeri Verkamo, "Discovery of Frequent Episodes in Event Sequences", Report C-1997-15, University of Helisnki, Department of Computer Science, 1997.

[12]Jussi Ahola, "Mining Sequential Patterns", Research Report TTE1-2001-10.

[13]Ramakrishnan Srikant and Rakesh Agrawal, "Mining Sequential Patterns: Generalizations and Performance Improvements," IBM Research Report RJ 1994, 1995.

[14]Oracle® Spatial User's Guide and Reference 10g Release 2 (10.2) B14255-01 Available at : http://download.oracle.com/docs/html/B10826_01/toc.htm

[15] Oracle® Spatial Topology and Network Data Models 10g Release 2 (10.2) B14256-02 Available at : http://download.oracle.com/docs/html/B10828_01/toc.htm